**Problem Statement:** The Objective is to predict the operating condition of a waterpoint for each record in the dataset. The three labels are provided in a separate dataset.

**Data Description:**

The dataset is available in the two files.

***waterpoint\_data\_values.csv***

**waterpoint\_data\_labels.csv**

**Water Point data values: (*waterpoint\_data\_values.csv*)**

Format: csv

Data Description:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| id | id of the waterpoint | Integer |
| amount\_tsh | Metric indicating total static head for pump; should be > 0 | double |
| date\_recorded | Date of data collection by survey company | string |
| funder | Name of organization that funded installation of pump | string |
| gps\_height | Altitude of pump | integer |
| installer | Name of organization that installed the pump | string |
| longitude | Longitudinal coordinate of pump | Double |
| latitude | Latitudinal coordinate of pump | Double |
| wpt\_name | Name assigned to given waterpoint | string |
| num\_private | No definition is available for this variable | integer |
| basin | Name of geographic water basin where pump is located | string |
| subvillage | Name of geographic subvillage where pump is located | string |
| region | Name of geographic region in Tanzania where pump is located | string |
| region\_code | Numeric ID for ‘region’ variable | integer |
| lga | Tanzania-specific geographic indicator of where pump is located | string |
| ward | Name of Tanzanian geographic ward where pump is located | string |
| population | Human population in immediate vicinty of pump | integer |
| public\_meeting | True/False indicator | boolean |
| recorded\_by | Name of the data collection / survey company | string |
| scheme\_management | Type of the organization responsible for management of pump | string |
| scheme\_name | Name of organization responsible for management of pump | string |
| permit | True/False indicating whether the pump has valid permit | boolean |
| construction\_year | The year the pump was installed | integer |
| extraction\_type | Method of extraction used at a given pump site. | string |
| extraction\_type\_group | Aggregation of extraction type categories. | string |
| extraction\_type\_class | Aggregation of extraction type group categories. | string |
| management | Name of method employed for management of a given pump. | string |
| management\_group | Possibly an aggregation of management categories. | string |
| payment | Categorical indicator of payment method required of pump users. | string |
| payment\_type | Appears to be a duplicate of payment categories. | string |
| water\_quality | Categorical indicator of water quality produced by pump. | string |
| quality\_group | Aggregation of water\_quality categories. | string |
| quantity | Categorical indicator of water quantity produced by pump. | string |
| quantity\_group | Aggregation of quantity categories. | string |
| source | Type of source of water for a given pump. | string |
| source\_type | Aggregation of source categories. | String |
| source\_type\_class | Aggregation of source\_type categories. | string |
| waterpoint\_type | The type of pump installed at a well site. | string |
| waterpoint\_type\_group | Aggregation of waterpoint\_type categories. | string |

**Waterpoint data labels: (****waterpoint\_data\_labels.csv)**

Format: csv

Data Description:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| id | id of the waterpoint | Integer |
| status\_group | Target Variable – Whether the water point is functional,Non functional or needs repair. | string |

Complete all the below tasks using Spark SQL and Spark ML:

**Task1:** Create RDD, Dataframe and schema for the two files. (10 marks)

**Task2:** Verify the row counts of the two dataFrames and check whether the counts are matching or not. (5 marks)

**Task3:** Verify first few rows of both the dataFrames and identify the common column name (key). (2 marks)

**Task4:** Create a DataFrame in cache . (2 marks)

**Task5:** Verify the row counts after the combined Data Frame. (3 marks)

Use the above combined Data Frame for further analysis.

**Task6:** Verify the distribution for the target variable using Spark SQL and visualize the distribution using plots. (5 Marks)

**Task7:** For the categorical variables **‘construction\_year’,’water\_quality’,** find the distribution of these variables with the target variable **‘status\_group’. Visualize** the data for these data frames. (5 Marks)

**Task8:** Create a new Dataframe by adding columns year,date and month from the existing date column on the dataframe. (3 Marks)

**Task9:** Change the fields to the following, cast ‘**public\_meeting**’ as String, cast ‘**permit**’ as String, cast ‘**gps\_height**’ as Double. (5 Marks)

**Task10:** Verify for the Nulls and NaNs in the data frame. (3 marks)

**Task11:** Impute the ‘**gps\_height**’ field with mean value for any missing values.

(Use from pyspark.ml.feature import Imputer). (5 marks)

**Task12:** Drop **‘id’,’record\_by’** and **‘date\_recorded’** from the dataframe. (2 marks)

**Task13:** Replace null values with the value ‘UNKNOWN’ for these fields **‘funder’,’installer’,’subvillage’,’public\_meeting’,’scheme\_management’,’scheme\_name’,’permit’**. (5 Marks)

**Task14:** Split the data as train and test sets and save the file. (5 marks)

**Task15:** Define the SparkMl Pipelines for the preprocessing stages as below.

(20 marks)

1. Combine all the numerical columns as vector (VectorAssembler for numerical columns)
2. Standardize the values in the above vector using StandardScaler available in pyspark.ml.feature
3. StringIndexer and OneHotEncoderEstimator for the categorical columns.
4. Combine all the above vectors created by OneHotEncoderEstimator as a single vector. (VectorAssembler for categorical col vectors)
5. Combine both the scaled numerical vector and combined categorical vectors (for the vectors obtained in ii and iv using VectorAssembler.
6. Convert the target variable into categories using various levels.