```
import pyspark
import pandas
from pyspark.sql.session import SparkSession
from pyspark.sql.context import SQLContext
from pyspark.sql.types import StructType
from pyspark.context import SparkContext, SparkConf

#spark = SparkSession.builder.master("spark://192.168.56.101:7077").appName("project").\
#config("spark.executor.instances",'2').\
#config("spark.executor.cores",'2').\
#getOrCreate()
#diseasedata = spark.read.csv('hdfs://master:9000/dataset/heart.csv',header=true);
#diseasedata.show(6)
```

```
sparkconf= SparkConf().setAppName('project').setMaster('spark://master:7077')
sc = SparkContext.getOrCreate(conf=sparkconf)
sc.stop()
```

```
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/03/29 23:55:55 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where
```

```
spark = SparkSession.builder.master("spark://master:7077").appName("project").getOrCreate()
disease_schema = StructType().add('Age','integer')\
                    .add('Sex','string')\
                    .add('ChestPainType','string')\
                    .add('RestingBP','double')\
                    .add('Cholesterol','double')\
                    .add('FastingBS','double')\
                    .add('RestingECG','string')\
                    .add('MaxHR','double')\
                    .add('ExerciseAngina','string')\
                    .add('Oldpeak','double')\
                    .add('ST_Slope','string')\
                    .add('HeartDisease','double')
```

```
disease_data = spark.read.csv('hdfs://master:9000/heart_disease_dataset/',header = True, inferSchema = True)
disease_data.show(5)
```

```
+--------+----------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+
|      ID|            Name|Age|Sex|Chestpain|RestingBP|Cholesterol|RestingECG|MaxHR|Exercise|HeartDisease|SusceptibilityIndex|
+--------+----------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+
|ID001601|   Fernanda Love| 61|  M|        2|      140|        284|         0|  123|       0|           1|                 15|
|ID001602|    Fidel Vaughn| 61|  M|        2|      120|        337|         0|   98|       0|           1|                 13|
|ID001603|   Finnley Grant| 74|  M|        1|      155|        310|         0|  112|       0|           1|                 16|
|ID001604|     Fiona Cross| 68|  M|        2|      134|        254|         0|  151|       0|           0|                  5|
|ID001605|Fisher Fernandez| 51|  F|        1|      114|        258|         1|   96|       1|           0|                  3|
+--------+----------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+
only showing top 5 rows
```

```
disease_data.columns
```

```
['ID',
 'Name',
 'Age',
 'Sex',
 'Chestpain',
 'RestingBP',
 'Cholesterol',
 'RestingECG',
 'MaxHR',
 'Exercise',
 'HeartDisease',
 'SusceptibilityIndex']
```

```
type(disease_data)
```

```
pyspark.sql.dataframe.DataFrame
```

```
pandas_disease_df = disease_data.toPandas()
#disease_data['HeartDisease'].value_counts()
```

```
pandas_disease_df['HeartDisease'].value_counts()
```

```
1    508
0    410
Name: HeartDisease, dtype: int64
```

```
pandas_disease_df.head(5)
```

	ID	Name	Age	Sex	Chestpain	RestingBP	Cholesterol	RestingECG	MaxHR	Exercise	HeartD
0	ID001601	Fernanda Love	61	M	2	140	284	0	123	0	
1	ID001602	Fidel Vaughn	61	M	2	120	337	0	98	0	
2	ID001603	Finnley Grant	74	M	1	155	310	0	112	0	
3	ID001604	Fiona	[illegible]	M	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	

```
#pandas_disease_df=pandas_disease_df.drop(index=0)
```

```
pandas_disease_df.head(2)
```

	ID	Name	Age	Sex	Chestpain	RestingBP	Cholesterol	RestingECG	MaxHR	Exercise	HeartDi
0	ID001601	Fernanda Love	61	M	2	140	284	0	123	0	
1	ID001602	Fidel	61	M	2	120	337	0	98	0	

```
pandas_disease_df['Age'].value_counts()
```

```
54    51
58    42
55    41
57    38
56    38
52    36
51    35
62    35
59    35
53    33
60    32
48    31
61    31
63    30
50    25
43    24
41    24
46    24
64    22
65    21
49    21
44    19
47    19
45    18
42    18
38    16
39    15
67    15
69    13
66    13
40    13
37    11
35    11
68    10
34     7
70     7
74     7
36     6
71     5
32     5
72     4
29     3
75     3
77     2
31     2
33     2
76     2
28     1
30     1
73     1
Name: Age, dtype: int64
```

```
pandas_disease_df['Sex'].value_counts()
```

```
M    725
F    193
Name: Sex, dtype: int64
```

```
pandas_disease_df['Sex'].value_counts()
```

```
M    725
F    193
Name: Sex, dtype: int64
```

```
pandas_disease_df['Chestpain'].value_counts()
```

```
1    496
2    203
3    173
5     46
Name: Chestpain, dtype: int64
```

```
pandas_disease_df['RestingBP'].value_counts()
```

```
120    132
130    118
140    107
110     58
150     55
      ...
113      1
80       1
129      1
92       1
98       1
Name: RestingBP, Length: 67, dtype: int64
```

```
pandas_disease_df['Cholesterol'].value_counts()
```

```
0      172
254     11
220     10
223     10
204      9
      ...
417      1
564      1
178      1
388      1
165      1
Name: Cholesterol, Length: 222, dtype: int64
```

```
#pandas_disease_df['ST_Slope'].value_counts()
```

```
df_rdd = disease_data.rdd
```

```
type(disease_data)
```

```
pyspark.sql.dataframe.DataFrame
```

```
df_rdd.count()
```

```
918
```

```
disease_data.show(5)
```

```
+--------+----------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+
|      ID|            Name|Age|Sex|Chestpain|RestingBP|Cholesterol|RestingECG|MaxHR|Exercise|HeartDisease|SusceptibilityIndex|
+--------+----------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+
|ID001601|   Fernanda Love| 61|  M|        2|      140|        284|         0|  123|       0|           1|                 15|
|ID001602|    Fidel Vaughn| 61|  M|        2|      120|        337|         0|   98|       0|           1|                 13|
|ID001603|   Finnley Grant| 74|  M|        1|      155|        310|         0|  112|       0|           1|                 16|
|ID001604|     Fiona Cross| 68|  M|        2|      134|        254|         0|  151|       0|           0|                  5|
|ID001605|Fisher Fernandez| 51|  F|        1|      114|        258|         1|   96|       1|           0|                  3|
+--------+----------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+
only showing top 5 rows
```

```
type(pandas_disease_df.iloc[2,2])
```

```
numpy.int32
```

```
pandas_disease_df['Age'] = pandas_disease_df['Age'].astype(int)
```

```
pandas_disease_df['Chestpain'] = pandas_disease_df['Chestpain'].astype(int)
```

```
pandas_disease_df['RestingBP'] = pandas_disease_df['RestingBP'].astype(int)
```

```
pandas_disease_df['RestingECG'] = pandas_disease_df['RestingECG'].astype(int)
```

```
pandas_disease_df['MaxHR'] = pandas_disease_df['MaxHR'].astype(int)
```

```
pandas_disease_df['Exercise'] = pandas_disease_df['Exercise'].astype(int)
```

```
pandas_disease_df['Heart Disease'] = pandas_disease_df['Heart Disease'].astype(int)
```

```
pandas_disease_df['Cholesterol'] = pandas_disease_df['Cholesterol'].astype(int)
```

```
pandas_disease_df['Susceptibility Index'] = pandas_disease_df['Cholesterol'].astype(int)
```

```
type(pandas_disease_df.iloc[2,2])
```

```
numpy.int64
```

```
pandas_disease_df.columns
```

```
Index(['ID', 'Name', 'Age', 'Sex', 'Chestpain', 'RestingBP', 'Cholesterol',
       'RestingECG', 'MaxHR', 'Exercise', 'HeartDisease',
       'SusceptibilityIndex'],
      dtype='object')
```

```
disease_data = spark.createDataFrame(pandas_disease_df)
```

```
/home/hduser/spark-3.3.2-bin-hadoop3/python/pyspark/sql/pandas/conversion.py:474: FutureWarning: iteritems is deprecated and will b
  for column, series in pdf.iteritems():
/home/hduser/spark-3.3.2-bin-hadoop3/python/pyspark/sql/pandas/conversion.py:486: FutureWarning: iteritems is deprecated and will b
  for column, series in pdf.iteritems():
```

```
required_features = ['Age',
                     'Chestpain',
                    'RestingBP',
                     'Cholesterol',
                     'RestingECG',
                     'MaxHR',
                     'Exercise',
                     'HeartDisease'
                    ]
```

```
from pyspark.ml.feature import VectorAssembler

assembler = VectorAssembler(inputCols=required_features, outputCol='features')

transformed_data = assembler.transform(disease_data)
```

```
(training_data, test_data) = transformed_data.randomSplit([0.8,0.2])
```

```
training_data.show(2)
```

```
+--------+-------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+--------
|      ID|         Name|Age|Sex|Chestpain|RestingBP|Cholesterol|RestingECG|MaxHR|Exercise|HeartDisease|SusceptibilityIndex|
+--------+-------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+--------
|ID001601|Fernanda Love| 61|  M|        2|      140|        284|         0|  123|       0|           1|                 15|[61.0,2.
|ID001602| Fidel Vaughn| 61|  M|        2|      120|        337|         0|   98|       0|           1|                 13|[61.0,2.
+--------+-------------+---+---+---------+---------+-----------+----------+-----+--------+------------+-------------------+--------
only showing top 2 rows
```

Unsupported Cell Type. Double-Click to inspect/edit the content.

```
from pyspark.ml.classification import LogisticRegression

lr = LogisticRegression(featuresCol = 'features', labelCol = 'SusceptibilityIndex')
lrModel = lr.fit(training_data)
lr_predictions = lrModel.transform(test_data)
```

```
lr_predictions.select('ID','Name','Age', 'Chestpain','RestingBP', 'RestingECG','MaxHR','Cholesterol', 'HeartDisease','SusceptibilityInde
```

```
+--------+----------------+---+---------+---------+----------+-----+-----------+------------+-------------------+
|      ID|            Name|Age|Chestpain|RestingBP|RestingECG|MaxHR|Cholesterol|HeartDisease|SusceptibilityIndex|
+--------+----------------+---+---------+---------+----------+-----+-----------+------------+-------------------+
|ID001605|Fisher Fernandez| 51|        1|      114|         1|   96|        258|           0|                  3|
|ID001611|   Forest Stuart| 62|        5|      135|         2|  137|        139|           0|                 10|
|ID001614| Francesco Huynh| 62|        3|      120|         1|   93|        254|           1|                 15|
|ID001615|  Francine Potts| 70|        1|      130|         1|  109|        322|           1|                 16|
|ID001618|  Franklin Gates| 64|        1|      128|         0|  105|        263|           0|                  3|
|ID001619| Franny Woodward| 74|        3|      120|         1|  121|        269|           0|                  6|
|ID001624|     Frieda Owen| 63|        1|      150|         1|  154|        407|           1|                 17|
|ID001627|     Fynn Massey| 44|        2|      140|         1|  180|        235|           0|                  6|
|ID001629|   Gael Griffith| 57|        1|      128|         1|  159|        303|           0|                  4|
|ID001641|    Gavyn Powers| 46|        1|      138|         1|  152|        243|           0|                  3|
+--------+----------------+---+---------+---------+----------+-----+-----------+------------+-------------------+
only showing top 10 rows
```

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

multi_evaluator = MulticlassClassificationEvaluator(labelCol = 'SusceptibilityIndex', metricName = 'accuracy')
print('Logistic Regression Accuracy:', multi_evaluator.evaluate(lr_predictions))
```

```
Logistic Regression Accuracy: 0.6352941176470588
```

```
lrModel.save('hdfs://master:9000/lrmodel')
```