



Big Data Parallel Programming

Final Project Report

ajmal8546@outlook.com

Ajmal Hussain

19930926-8218

Contents

1	State of Problem	2
2	Introduction	2
3	Discussion.....	3
3.1	Google Cloud Platform.....	3
3.1.1	Creating Cluster.....	3
3.1.2	Creating Firewall Rule	3
3.1.3	Static ip address	4
3.1.4	Storage Bucket	4
3.2	Configure Jupyter Notebook.....	5
3.3	Pyspark.....	5
3.4	Exploratory Analysis.....	5
3.4.1	Showing The Data	6
3.4.2	Missing Values.....	6
3.4.3	Statistics of Data	7
3.4.4	Data filtering	7
3.4.5	Graphical representation	8
3.4.6	String Indexer	8
3.4.7	Vector Assembler	9
3.4.8	Stand Scaler.....	9
3.4.9	Split the dataset	10
3.4.10	Data Balancing	10
3.5	Machine Learning.....	11
3.5.1	Logistic Regression	11
3.5.2	Decision Tree.....	12
3.5.3	Random Forest.....	12
3.5.4	Gradient Boost Tree Classifier.....	13
3.5.5	Linear SVS.....	14
3.6	Model Selection	15
3.7	Creating Output Link on Jupyter Notebook.....	17
3.8	Saving output on GCP Bucket	17
3.9	Job submission	17
4	Problems and solution on GCP	18
5	Conclusion.....	19

1 State of Problem

In This project we have to predict whether an individual's income will be greater than \$50,000 per year or less than \$50000 per year based on several attributes from the census data that is collected from Kaggle US adult census dataset.

Keywords: Google cloud platform, Machine learning, Decision Tree, Gradient Boos Tree Classification, Logistic Regression, Random Forest,

2 Introduction

The obvious difference of wealth and income is a huge concern, especially in the United States. The principle of universal moral equality guarantees the improvement of the financial stability and sustainable development. The different countries are trying their best way to overcome on this problem and giving an optimal solution. I will contribute to this problem by performing some machine learning algorithm.

This is a supervised machine learning classification task. In supervised Machine learning we have given data attribute with labeled class and we have to train a model by using different algorithms and predict the data that may have lack of class label. In This task we have a census income dataset with several attributes. The dataset contains 48,842 entries and 14 attributes. Each entry contains the information about an individual. This task is done by using pyspark on GCP. I used cluster with one master node and 2 worker nodes. A firewall also created to use the ip address and port number 8888 to access the cluster node with specific ip address and port number. the cluster node uses the static ip address rather than the dynamic. After this loaded the data from gcp bucket with pyspark and perform some exploratory analysis on the data. To be very first I check the schema of the data with print schema and then check the missing or unknown value in the data. I drop out the unknown value and then perform the filtering on the data to check the association of the attribute with income. There are some categorical attributes that are converted to numerical by using string indexer. After this I used the vector assembler to put all the numeric attribute into one column is call features. In addition, I used the stand scaler to scale all the values in features column and saved output in scaled features column. Then created the pipeline that means how the stages will be performed on the previous steps and prepared a data pipeline. After the pipeline, the data is ready for machine learning algorithms. I performed five different algorithm that are Logistic Regression, Decision Tree, Random Forest, Gradient Boost Tree Classifier and Linear SVC.I used these algorithms and perform the hyper parameter tuning of each algorithm. Each algorithm gives the different accuracy and based on accuracy I evaluated the algorithm and choose a best model to predict the test data. After this the output of the test data saved in a csv file on GCP bucket that user can download the output file as well. I also generated the http link on the jupyter notebook the user can also download the file directly from the notebook. After completing the code in notebook, I created a pyspark job on GCP cluster with my python file and submit the job successfully. The output can be seen on the GCP and, we can monitor the job as well as can see the log of the job. I have also tried the IBM Watson for this project but according to some limitation I switch on to GCP rather than IBM Watson. One of the major problems of IBM Watson is that We can perform limited prediction on IBM Watson. In the next session I will briefly explain the methodology, discussion and results as well .

3 Discussion

In this part the whole project methodology will be explained that means the flow of project including GCP cluster ,configuration ,creating firewall rule, set the static ip address, the data exploratory analysis , Statistics ,Data cleaning, Data filtering , String indexer, Vector assembler, Stand scaler, Data pipeline and train model with five different binomial classification algorithm and measure the accuracy . All algorithm evaluated on the base of ROC and select the best model and make prediction with these model and then submit a pyspark job then save the output file on GCP Bucket that user can download the output file as well .

3.1 Google Cloud Platform

GCP provide different features like clustering, creating VM instance, storage, private network, firewall rule and much other features. I am using the GCP for creating cluster, private network, firewall rule, storage and job submission

3.1.1 Creating Cluster

To be very first I created the GCP account with 300\$ free credit with 365 days. After this I created a project with the name of BDPPPProject .In addition, I click on menu on dashboard and go to Dataproc and click on cluster for creating the cluster with one master node and 2 worker node with name of bdppcluster .the following screenshot can be seen what I have created :-

Google Cloud Platform | BDPPPProject | Search products and resources

Dataproc | Cluster details | SUBMIT JOB | REFRESH | DELETE | VIEW LOGS

bdppcluster

For PD-Standard without local SSDs, we strongly recommend provisioning 1TB or larger to ensure consistently high I/O performance. See <https://cloud.google.com/compute/docs/disks/performance> for information on disk I/O performance.

MONITORING | JOBS | VM INSTANCES | CONFIGURATION | WEB INTERFACES

Filter instances

Name ↑	Role	SSH
bdppcluster-m	Master	SSH
bdppcluster-w-0	Worker	
bdppcluster-w-1	Worker	

Equivalent [REST](#)

3.1.2 Creating Firewall Rule

After completed the cluster creation I need to create a firewall rule. I click on option and go the vps network then click on firewall rule and create a firewall rule with name of firewal for setting up the specific tcp port number 8888 and set the ip address range 0.0.0.0/0. So I can access my virtual machine instance with ip address and specific port number. I also allow the http traffic and allow the access to all project .following is the screen shot of firewall rule you can see there are some default firewall rule but I created a firewall rule with the name of firewall

Name	Type	Targets	Filters	Protocols/ports	Action	Priority	Network	Logs	Hit count
default-allow-http	Ingress	http-server	IP ranges: 0.0.0.0/0	tcp:80	Allow	1000	default	Off	
default-allow-https	Ingress	https-server	IP ranges: 0.0.0.0/0	tcp:443	Allow	1000	default	Off	
firewall	Ingress	Apply to all	IP ranges: 0.0.0.0/0	tcp:8888	Allow	1000	default	Off	

3.1.3 Static ip address

After the creation of cluster and firewall rule I click on external ip address where I have to set the ip address static so I can reserve the external ip address for my virtual machine instances if don't set it to static then may the ip address change after some while and I need to access with new ip address so that's why I set the ip address static and it will reserve for my VM instances. You can see the following screen shot with all three instances that mean one master node and 2 worker nodes with their ip static ip address:

Name	External address	Region	Type	Version	In use by	Network tier	Labels
bddp1	35.228.21.54	europe-north1	Static	IPv4	VM instance bddpcluster-m (Zone europe-north1-a)	Premium	CHANGE
bddp3	35.228.92.107	europe-north1	Static	IPv4	VM instance bddpcluster-w-1 (Zone europe-north1-a)	Premium	CHANGE
bddp2	35.228.53.172	europe-north1	Static	IPv4	VM instance bddpcluster-w-0 (Zone europe-north1-a)	Premium	CHANGE

3.1.4 Storage Bucket

Now I created a bucket on google cloud platform with the name of ajmal1 where I can upload my dataset and store the output file on bucket where a user can download the output file as well. The following screen shot contain the information of storage bucket where you can see the dataset and other file as well

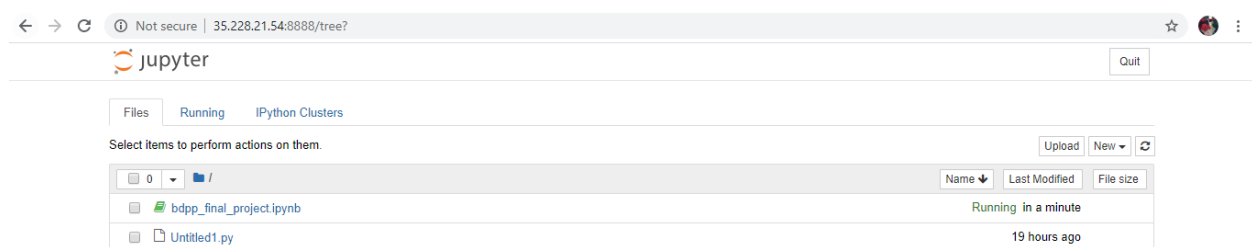
Name	Size	Type	Storage class	Last modified	Public access	Encryption	Retention expiry date	Holds
bddp_final_proj.py	20.31 KB	text/plain	Standard	21/05/2020, 04:20:35 UTC+2	Not public	Google-managed key	--	None
data_frame.csv	5.08 MB	application/vnd.ms-excel	Standard	13/05/2020, 09:31:52 UTC+2	Not public	Google-managed key	--	None
output11.csv	7.29 MB	application/octet-stream	Standard	21/05/2020, 05:27:51 UTC+2	Not public	Google-managed key	--	None
project.py	20.41 KB	text/plain	Standard	21/05/2020, 05:13:33 UTC+2	Not public	Google-managed key	--	None

3.2 Configure Jupyter Notebook

After setting up the cluster ,firewall rule, external ip address and storage bucket. I configure the jupyter notebook with the top port number on master node . I opened the jupyter configuration file and set the required field as ip address and port number. I give the command “ vi ~/.jupyter/jupyter_notebook_config.py”

```
# Configuration file for jupyter-notebook.
c = get_config()
c.NotebookApp.ip = '*'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 8888
#-----
# Application(SingletonConfigurable) configuration
#-----
```

After setting up the configuration I run the command “jupyter notebook “ on master node and access the jupyter notebook with external ip address and tcp port 8888 write down the 35.228.21.54:8888 and You can access the jupyter notebook



3.3 Pyspark

To be very first in jupyter notebook I created a pyspark session and load the data from the GCP bucket into the jupyter notebook “ traindata = spark.read.load("gs://ajmal1/data_frame.csv",format="csv", sep=";", inferSchema=True,schema=schema, header=True) “

3.4 Exploratory Analysis

Firstly, I check out the length of data which is 48842 rows and 15 columns. I also check the schema of the data with the command of printSchema. You can see the following screen shot

```
In [3]: #rows and col in dataset
length=traindata.count(),len(traindata.columns)
print("length of data is :",length)
length of data is : (48842, 15)

In [4]: #checking the schema of the data
print("Schema of the data")
traindata.printSchema()

Schema of the data
root
 |-- age: integer (nullable = true)
 |-- workclass: string (nullable = true)
 |-- final_weight: integer (nullable = true)
 |-- education: string (nullable = true)
 |-- educational_num: integer (nullable = true)
 |-- marital_status: string (nullable = true)
 |-- occupation: string (nullable = true)
 |-- relationship: string (nullable = true)
 |-- race: string (nullable = true)
 |-- sex: string (nullable = true)
 |-- capital_gain: integer (nullable = true)
 |-- capital_loss: integer (nullable = true)
 |-- hours_per_week: integer (nullable = true)
 |-- native_country: string (nullable = true)
 |-- income: string (nullable = true)
```

Activate Windows

Here we can see that there are 15 columns and 48842 rows in the datasets The Dataset the Census Income dataset has 48,842 entries. Each entry contains the following information about an individual:

- age: the age of an individual
- workclass: a general term to represent the employment status of an individual
- fnlwgt: final weight. In other words, this is the number of people the census believes the entry represents
- education: the highest level of education achieved by an individual.
- education_num: the highest level of education achieved in numerical form
- maritalstatus: marital status of an individual
- occupation: the general type of occupation of an individual
- relationship: represents what this individual is relative to others.
- race: Descriptions of an individual's race
- sex: the biological sex of the individual
- capitalgain: capital gains for an individual
- capitalloss: capital loss for an individual
- hoursperweek: the hours an individual has reported to work per week
- nativecountry: country of origin for an individual
- the label: whether or not an individual makes more than \$50,000 annually.

3.4.1 Showing The Data

From the following screenshot you can see the first 5 rows of dataset.

```
#showing the first 5 rows of Data
print("First five rows of data")
traindata.show(5)
```

First five rows of data

	age	workclass	final_weight	education	educational_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	native_country	income
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	0	0	40	United-States	<=50K
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	50	United-States	<=50K
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	0	0	40	United-States	>50K
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	7688	0	40	United-States	>50K
4	18	?	103497	Some-college	10	Never-married	?	Own-child	White	Female	0	0	30	United-States	<=50K

only showing top 5 rows

3.4.2 Missing Values

The dataset contain “?” value which means the null value are unknown value I drop out these rows that contain “?” value. here you can show the data without “?” value .

```
#dropout the unknown value
new_df = traindata.filter((traindata.workclass != '?') & (traindata.occupation != '?') & (traindata.native_country != '?'))
new_df.show(5)
```

	age	workclass	final_weight	education	educational_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	native_country	income
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	0	0	40	United-States	<=50K
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	50	United-States	<=50K
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	0	0	40	United-States	>50K
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	7688	0	40	United-States	>50K
4	34	Private	198693	10th	6	Never-married	Other-service	Not-in-family	White	Male	0	0	30	United-States	<=50K

only showing top 5 rows

3.4.3 Statistics of Data

Here you can see the statistics of some attribute that mean the count ,minimum , maximum , mean standard deviation .

```
#statistics of data
new_df.describe(['age', 'capital_gain', 'hours_per_week', 'capital_loss', 'final_weight']).show()
```

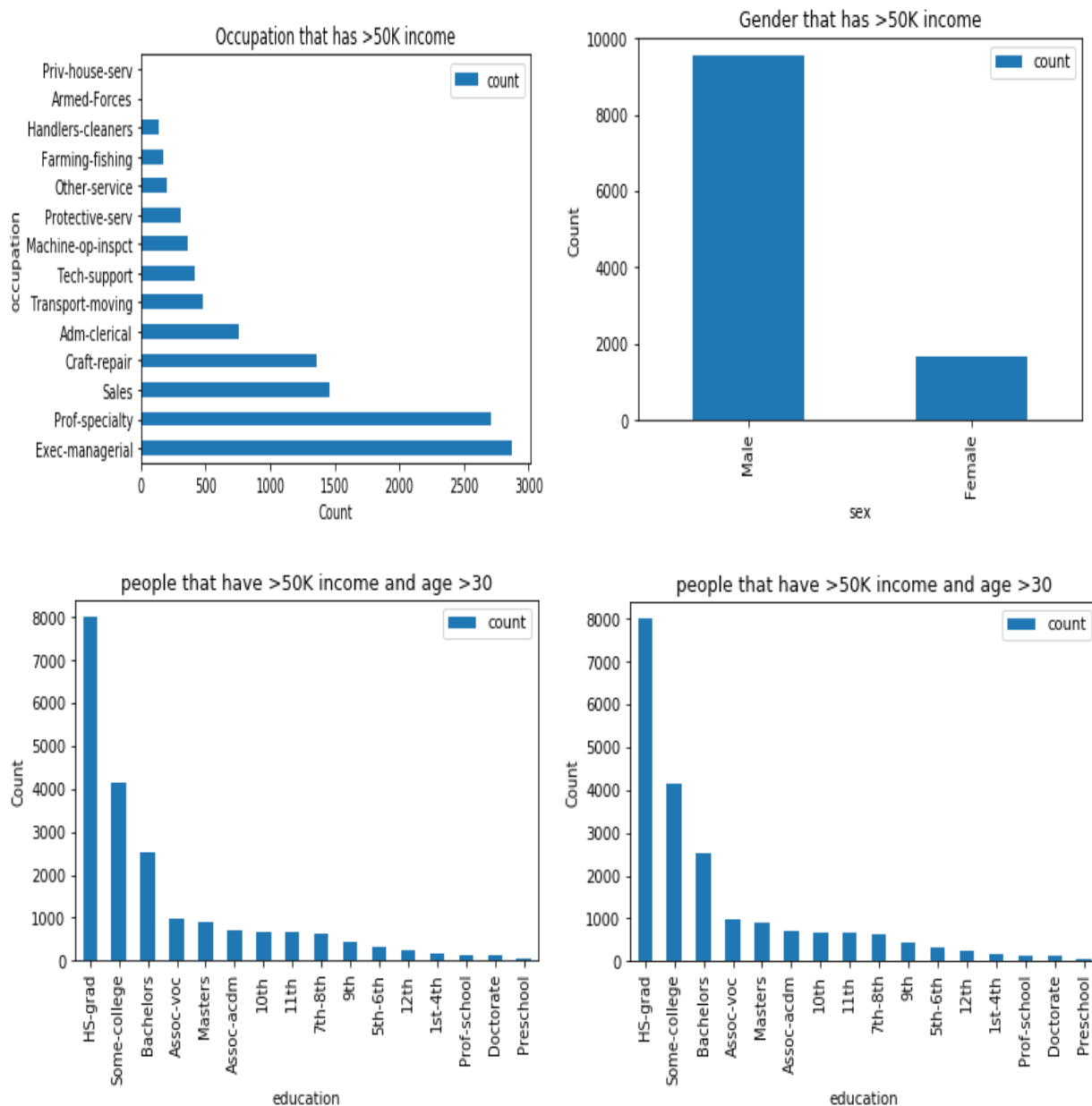
summary	age	capital_gain	hours_per_week	capital_loss	final_weight
count	45222	45222	45222	45222	45222
mean	38.547941267524656	1101.4303436380524	40.93801689443191	88.595418159303	189734.7343107337
stddev	13.217870219055532	7506.43008374525	12.007508230033423	404.95609205896506	105639.19513422064
min	17	0	1	0	13492
max	90	99999	99	4356	1490400

3.4.4 Data filtering

Here you can see the different filter data.

>50k \$ income grouped by occupation		<50K \$ income and capital gain >100K and age >30		>50k \$ income	
occupation	count	occupation	count	sex	count
Craft-repair	2867	Craft-repair	204	Male	9539
Exec-managerial	2704	Adm-clerical	185	Female	1669
Prof-specialty	1455	Prof-specialty	164		
Sales	1355	Exec-managerial	143		
Craft-repair	756	Sales	142		
Adm-clerical	478	Other-service	118		
Transport-moving	411	Machine-op-inspct	112		
Tech-support	365	Transport-moving	73		
Machine-op-inspct	307	Handlers-cleaners	58		
Protective-serv	196	Farming-fishing	57		
Other-service	172	Tech-support	55		
Farming-fishing	135	Protective-serv	30		
Handlers-cleaners	4	Priv-house-serv	8		
Armed-Forces	3				
Priv-house-serv					

3.4.5 Graphical representation



3.4.6 String Indexer

I have categorical attributes in dataset I have to convert it into the numerical form by string indexer .the String indexer is used to convert the categorical col to numerical col.i have categorical columns are "workclass", "marital_status", "occupation", "relationship", "race", "sex", "native_country" and incom .the following screen shot is creating the numerical col from categorical col .

```

from pyspark.ml.feature import StringIndexer
from pyspark.ml.feature import OneHotEncoderEstimator
categoricalColumns = ["workclass", "marital_status", "occupation", "relationship", "race", "sex", "native_country"]
stages = [] # stages in our Pipeline
for categoricalCol in categoricalColumns:
    # Category Indexing with StringIndexer
    stringIndexer = StringIndexer(inputCol=categoricalCol, outputCol=categoricalCol + "Index")
    stages += [stringIndexer]
#stages
#stringIndexer.show(5)

#convert the Label into numeric
label_stringIdx = StringIndexer(inputCol="income", outputCol="label")
#new_df = label_stringIdx.fit(new_df).transform(new_df)
|
stages += [label_stringIdx]
#new_df.show(1)

```

3.4.7 Vector Assembler

Vector assembler is used to combine all the attributes in one feature col and this feature or scaled features col is used for training the model as well. you can see the following output of col feature

```

showing dataset with feature col after assembler
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|          features|label|age|final_weight|educational_num|capital_gain|capital_loss|hours_per_week|workclass|marital_sta|
tus|      occupation| relationship| race| sex|native_country|income|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| [0.0,1.0,6.0,2.0,...| 0.0| 25| 226802| 7|
ied|Machine-op-inspct| Own-child|Black|Male| United-States| <=50K| 0| 0| 40| Private| Never-marr
| (13,[2,7,8,9,12],...| 0.0| 38| 89814| 9|
use| Farming-fishing| Husband|White|Male| United-States| <=50K| 0| 0| 50| Private|Married-civ-spo
| (13,[0,2,7,8,9,12,...| 1.0| 28| 336951| 12|
use| Protective-serv| Husband|White|Male| United-States| >50K| 0| 0| 40| Local-gov|Married-civ-spo
| (13,[2,4,7,8,9,10,...| 1.0| 44| 160323| 10|
use|Machine-op-inspct| Husband|Black|Male| United-States| >50K| 7688| 0| 40| Private|Married-civ-spo
| (13,[1,2,3,7,8,9,...| 0.0| 34| 198693| 6|
ied| Other-service|Not-in-family|White|Male| United-States| <=50K| 0| 0| 30| Private| Never-marr
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows

```

3.4.8 Stand Scaler

Stand scaler is used for scale the values in feature col. If there are values that have much difference then the stand scaler scale all the values and generate scaled features col . You can see the following snap

```

showing dataset with scaled feature after scalling
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|          features|scaledFeatures|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| [0.0,1.0,6.0,2.0,...| [-0.5024394581349...|
| (13,[2,7,8,9,12],...| [-0.5024394581349...|
| (13,[0,2,7,8,9,12,...| [1.03413436858976...|
| (13,[2,4,7,8,9,10,...| [-0.5024394581349...|
| (13,[1,2,3,7,8,9,...| [-0.5024394581349...|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows

```

3.4.9 Split the dataset

Now the data is almost ready here we will check that split data count that how much is data for testing and how much data for training you can see the following screen shot to see the size of test and train data

```
#split the data
### Randomly split data into training and test sets. set seed for reproducibility
(trainingData, testData) = testing.randomSplit([0.7, 0.3], seed=100)
print("Training Data",trainingData.count())
print("Testing Data",testData.count())
```

```
Training Data 31676
Testing Data 13546
```

Here we can see that 31676 for training and 13546 for testing the data i.e 70% for training and 30% for testing now we will check that data is balance or not

```
#people that have $>50k incom and <=50k
higher=trainingData.filter(trainingData['income']=='>50K').count()
lower=trainingData.filter(trainingData['income']=='<=50K').count()
#trainingData.groupBy("income").count().show()
print("people that have the income >50k per year:",higher)
print("people that have the income <=50k per year:",lower)
print("Data is imbalance so we need to balance the data")
```

```
people that have the income >50k per year: 7887
people that have the income <=50k per year: 23789
Data is imbalance so we need to balance the data
```

I have 7887 people that have >\$50000 and 23789 have <=\$500000 which mean data is imbalanced so I need data balancing

3.4.10 Data Balancing

I need under sampling for balancing the data that mean the 7887 number of people that have >\$50000 should be the around about the people 23789 that have <=\$500000 .ypu can see the following screenshot of doing under sampling as well :-

```
print("After downsampling")
higher=balanced_df.filter(balanced_df['income']=='>50K').count()
lower=balanced_df.filter(balanced_df['income']=='<=50K').count()
print("people that have the income >50k per year:",higher)
print("people that have the income <=50k per year:",lower)
#print(testData.count())
```

```
After downsampling
people that have the income >50k per year: 7887
people that have the income <=50k per year: 5820
```

Now you can see that the data is balanced now we can train the different model and predict the data.

3.5 Machine Learning

machine learning is basically performing some rules on the dataset and make prediction of test data. This is supervised machine learning task and in supervised machine learning the data set always given with the label class. so I have data and I perform some steps on data. Now the data has been ready for the processing of machine learning. Now I will perform different number of algorithm and will find the ROC that will help to evaluate the algorithm.

3.5.1 Logistic Regression

Logistic regression is used for classification task to predict the label class by using the given information. I train the logistic regression model with training data and then predict the test data. You can see the output of logistic regression algorithm with label, prediction, income, age, native country. we can add more feature in output if we want but now I only showing the selected attributes. the following screenshot can be seen. Here you can see the accuracy of logistic regression without hyper parameter tuning

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

Output Configuration

☐ Line wrapping

20/05/21 03:16:31 INFO breeze.optimize.LBFGS: Converged because max iterations reached
prediction with col label, prediction, age, native country and income

```
+-----+-----+-----+-----+
|label|prediction|age|native_country|income|
+-----+-----+-----+-----+
| 0.0|      1.0| 28| United-States| <=50K|
| 1.0|      1.0| 33| United-States| >50K|
| 0.0|      0.0| 33| United-States| <=50K|
| 1.0|      1.0| 36| United-States| >50K|
| 0.0|      1.0| 40| United-States| <=50K|
+-----+-----+-----+-----+
```

only showing top 5 rows

Logistic regression ROC 0.8788368427002137

I used the ParamGridBuilder for parameter tuning. it gives me the best parameters and I trained the model with these credentials you can see there is slight difference between the accuracy. You can see the accuracy after parameter tuning following: -

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

Output Configuration

☐ Line wrapping

```
20/05/21 03:20:00 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.482458 (rel: 0.000746) 0.00586644
20/05/21 03:20:01 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/05/21 03:20:01 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.482382 (rel: 0.000157) 0.00106681
20/05/21 03:20:01 INFO breeze.optimize.OWLQN: Step Size: 1.000
20/05/21 03:20:01 INFO breeze.optimize.OWLQN: Val and Grad Norm: 0.482379 (rel: 7.25e-06) 0.000578394
20/05/21 03:20:01 INFO breeze.optimize.OWLQN: Converged because max iterations reached
After parameter tuning Lr ROC 0.8733750510484185
```

3.5.2 Decision Tree

After the Logistic regression I implemented the Decision tree and make prediction based on this model. I train this model with max depth=3 and num nodes= 11 and you can see the following screenshot for the out put of decesion tree and also you can see the ROC =0.78 of the decision tree as well

```
job-c103ca69
Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:
Output Configuration
Line wrapping
Equivalent command line
After parameter tuning Lr ROC: 0.8733750510484105
numNodes in Decsionn Tree = 11
depth in Decsion Tree= 3
Output of DT algorithm
+-----+-----+-----+
|label|prediction|      probability|age|
+-----+-----+-----+
| 0.0|      0.0|[0.82617303504256...| 28|
| 1.0|      0.0|[0.82617303504256...| 33|
| 0.0|      0.0|[0.82617303504256...| 33|
| 1.0|      0.0|[0.82617303504256...| 36|
| 0.0|      0.0|[0.82617303504256...| 40|
+-----+-----+-----+
only showing top 5 rows
Decision tree ROC 0.788900102997392
```

After this I used ParamGrid and Crossvalidator for parameter tuning .the param grid give me the best parameter for this model and crossvalidator split the data as well.now I train the model with the best parameters you can see the following screenshot the decision tree after the parameter tuning and it increase the value of ROC =0.8128 which is better the model trained before.

```
job-c103ca69
Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:
Output Configuration
Line wrapping
Equivalent command line
numNodes = 503
depth = 10
dt_hyper_model ROC: 0.8128513580821601
+-----+-----+-----+
|label|prediction|      probability|age|
+-----+-----+-----+
| 0.0|      0.0|[0.90047393364928...| 28|
| 1.0|      1.0|[0.03508771929824...| 33|
| 0.0|      0.0|[0.98822836962919...| 33|
| 1.0|      1.0|[0.03508771929824...| 36|
| 0.0|      1.0|[0.35471698113207...| 40|
+-----+-----+-----+
only showing top 5 rows
```

3.5.3 Random Forest

In This part I used the Random forest algorithm to train the model and make prediction based on this model. The model is evaluated by the binary classification evaluator. In the following screenshot you can see the evaluated Roc of this model

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

Output Configuration

☐ Line wrapping

label	prediction	probability	age
0.0	0.0	[0.76083136318804...	28
1.0	0.0	[0.50867942808311...	33
0.0	0.0	[0.81884497085259...	33
1.0	1.0	[0.40129838436650...	36
0.0	1.0	[0.40129838436650...	40

only showing top 5 rows

Random Forest ROC 0.8926806865856525

After this again I tune the parameter and made model based on these best credentials. And make prediction based on this model. You can see the out in following screenshot as well

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

Output Configuration

☐ Line wrapping

label	prediction	probability	age
0.0	0.0	[0.71482621590813...	28
1.0	1.0	[0.46523267821853...	33
0.0	0.0	[0.77768837739877...	33
1.0	1.0	[0.39225283179648...	36
0.0	1.0	[0.43725902887041...	40

only showing top 5 rows

Rf_hyper ROC 0.9028957876052465

3.5.4 Gradient Boost Tree Classifier

The fourth algorithm that I have implemented on this dataset is gradient boost tree classifier. the gradient boost tree classifier is used for binomial classification and I train the model with GBTC and made prediction based on this model. I evaluated the model based on ROC you can see the following screen shot for output of the GBTC.

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

Output Configuration

☐ Line wrapping

age	label	rawPrediction	prediction	probability
28	0.0	[0.90374032467220...	0.0	[0.85905711219807...
33	1.0	[-0.5203579257339...	1.0	[0.26101189351055...
33	0.0	[1.01274938190622...	0.0	[0.88344839703806...
36	1.0	[-0.6218952477054...	1.0	[0.22377688437729...
40	0.0	[-0.4940102041140...	1.0	[0.27130326136549...
42	0.0	[-0.1394899229710...	1.0	[0.43070389814323...
42	1.0	[-0.4819674767176...	1.0	[0.27609104375507...
45	0.0	[0.00481205929246...	0.0	[0.50240601107513...
56	0.0	[0.01623950985677...	0.0	[0.50811904121853...
20	0.0	[1.16328856057226...	0.0	[0.91105435607651...

only showing top 10 rows

GBT ROC 0.9064958987047095

After this I have done the hyper parameter tuning and train the model based on these best parameters that is received from paramgrid and made prediction with this model. You can see the Roc of this model after parameter tuning. There is slight difference between the ROC.

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

Output Configuration

☐ Line wrapping

age	label	rawPrediction	prediction	probability
28	0.0	[0.90374032467220...	0.0	[0.85905711219807...
33	1.0	[-0.5203579257339...	1.0	[0.26101189351055...
33	0.0	[1.01274938190622...	0.0	[0.88344839703806...
36	1.0	[-0.6218952477054...	1.0	[0.22377688437729...
40	0.0	[-0.4940102041140...	1.0	[0.27130326136549...
42	0.0	[-0.1394899229710...	1.0	[0.43070389814323...
42	1.0	[-0.4819674767176...	1.0	[0.27609104375507...
45	0.0	[0.00481205929246...	0.0	[0.50240601107513...
56	0.0	[0.01623950985677...	0.0	[0.50811904121853...
20	0.0	[1.16328856057226...	0.0	[0.91105435607651...

only showing top 10 rows

gbt_hyper ROC 0.9064983103409463

3.5.5 Linear SVS

The last algorithm I implemented on this data is linear SVS with max iteration = 10. I train the model according to the parameters and make prediction with this model. I evaluated this model based on the ROC. You can see the following screen shot for the Linear SVS output as well.

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

Output Configuration

☐ Line wrapping

```
+-----+-----+-----+-----+
|age|label|rawPrediction|prediction|
+-----+-----+-----+-----+
| 28| 0.0|[-0.3354399751693...| 1.0|
| 33| 1.0|[-0.9220932089910...| 1.0|
| 33| 0.0|[0.31099381987897...| 0.0|
| 36| 1.0|[-1.0414864969266...| 1.0|
| 40| 0.0|[-0.7345523169963...| 1.0|
| 42| 0.0|[-0.8992195695730...| 1.0|
| 42| 1.0|[-2.2749527398024...| 1.0|
| 45| 0.0|[-0.7050122553568...| 1.0|
| 56| 0.0|[-0.6499585977661...| 1.0|
| 20| 0.0|[1.23328907293674...| 0.0|
+-----+-----+-----+-----+
```

only showing top 10 rows

LVC ROC 0.8628705536803618

After this I have done the hyper parameter tuning with ParamGrid and Cross validator and the train model according to the credential that is received from Paramgrid and Cross validator. You can see the output of this model below as well.

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

Output Configuration

☐ Line wrapping

```
+-----+-----+-----+-----+
|age|label|rawPrediction|prediction|
+-----+-----+-----+-----+
| 28| 0.0|[-0.3354399751693...| 1.0|
| 33| 1.0|[-0.9220932089910...| 1.0|
| 33| 0.0|[0.31099381987897...| 0.0|
| 36| 1.0|[-1.0414864969266...| 1.0|
| 40| 0.0|[-0.7345523169963...| 1.0|
| 42| 0.0|[-0.8992195695730...| 1.0|
| 42| 1.0|[-2.2749527398024...| 1.0|
| 45| 0.0|[-0.7050122553568...| 1.0|
| 56| 0.0|[-0.6499585977661...| 1.0|
| 20| 0.0|[1.23328907293674...| 0.0|
+-----+-----+-----+-----+
```

only showing top 10 rows

LVC Hyper ROC 0.8628705536803694

3.6 Model Selection

I have implemented the five different model on this dataset and now its time to choose the best model that have trained with different parameters. I selected the best model based on the ROC. I have the ROC of all the model so I can choose the model that have the best ROC score. Predict the test data with this best model and save the output file on GCP bucket. Following table is containing the information of all five-model including the ROC before the parameters tuning and after the parameters tuning as well and I will select the best model based on the ROC.

ROC Table of algorithms

No#	Name	ROC Before Parameter Tuning	ROC After parameter tuning
1	Logistic Regression	0.8799046772930779	0.8733750510484185
2	Decision Tree	0.788900102997392	0.8128513580821601
3	Random Forest	0.8926806865856525	0.9028957876052465
4	Gradient Boost Tree Classification	0.9064958987047095	0.9064983103409463
5	Linear SVS	0.8628705536803618	0.8628705536803694

We can see that from the table that all the model has different ROC and the Gradient Boost Tree Classifier has the maximum ROC score. I chosed the best model Gradient Boost Tree Classifier for the final prediction as it has the maximum score based on ROC. You can see the final prediction output with best model Gradient Boost Tree Classifier following: -

✓ job-c103ca69

Start time: 21 May 2020, 05:14:03 Elapsed time: 14 min 4 sec Status:

[Output](#) [Configuration](#)

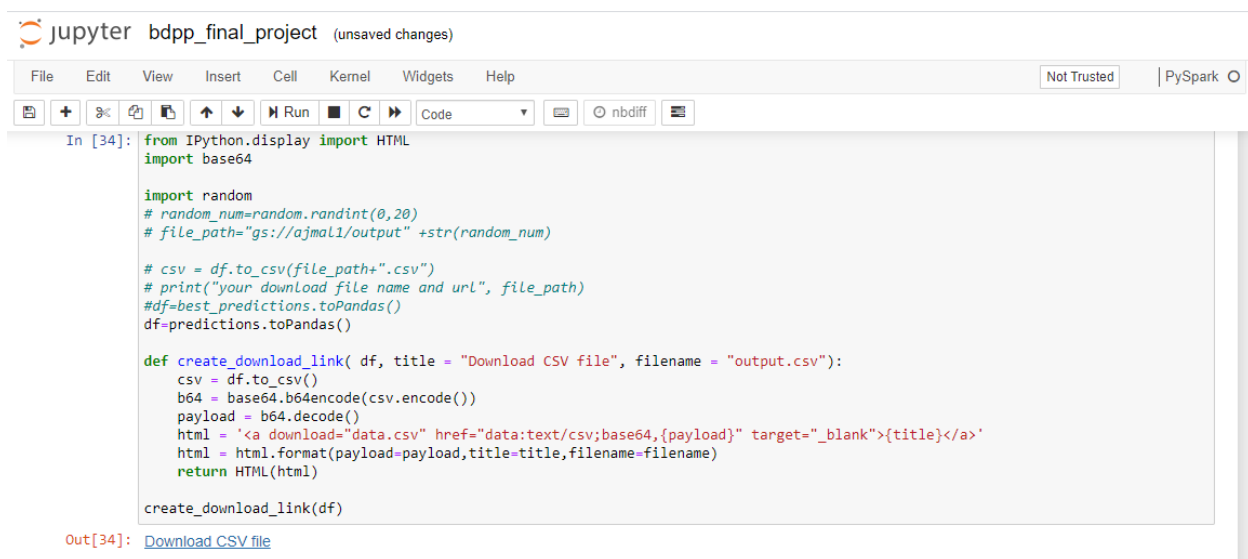
☐ Line wrapping

```
LVC Hyper ROC 0.8628705536803694
[[LogisticRegressionModel: uid = LogisticRegression_a9aebb683842, numClasses=2,
All Model with Accuracy [[DecisionTreeClassificationModel (uid=DecisionTreeClassificationModel_d9cf700d6884, 0.9064983103409463)
the best model is [CrossValidatorModel_d9cf700d6884, 0.9064983103409463]
+---+-----+-----+-----+
|age|label|      rawPrediction|prediction|
+---+-----+-----+-----+
| 28|  0.0|[0.90374032467220...|      0.0|
| 33|  1.0|[-0.5203579257339...|      1.0|
| 33|  0.0|[1.01274938190622...|      0.0|
| 36|  1.0|[-0.6218952477054...|      1.0|
| 40|  0.0|[-0.4940102041140...|      1.0|
| 42|  0.0|[-0.1394899229710...|      1.0|
| 42|  1.0|[-0.4819674767176...|      1.0|
| 45|  0.0|[0.00481205929246...|      0.0|
| 56|  0.0|[0.01623950985677...|      0.0|
| 20|  0.0|[1.16328856057226...|      0.0|
+---+-----+-----+-----+
only showing top 10 rows

ROC 0.9064983103409462
```

3.7 Creating Output Link on Jupyter Notebook

Finally, I have chosen the best model and made prediction with best model. Now I will create a hyper link in jupyter notebook where a user can download the output of prediction by clicking on the link on jupyter notebook. You can see the following screen shot for creating the hyper link for downloading the output file.



```

jupyter bdpf_final_project (unsaved changes)
File Edit View Insert Cell Kernel Widgets Help Not Trusted PySpark
In [34]: from IPython.display import HTML
import base64

import random
# random_num=random.randint(0,20)
# file_path="gs://ajmal1/output" +str(random_num)

# csv = df.to_csv(file_path+".csv")
# print("your download file name and url", file_path)
# df=best_predictions.toPandas()
df=best_predictions.toPandas()

def create_download_link( df, title = "Download CSV file", filename = "output.csv"):
    csv = df.to_csv()
    b64 = base64.b64encode(csv.encode())
    payload = b64.decode()
    html = '<a download="data.csv" href="data:text/csv;base64,{payload}" target="_blank">{title}</a>'
    html = html.format(payload=payload,title=title,filename=filename)
    return HTML(html)

create_download_link(df)

Out[34]: Download CSV file

```

3.8 Saving output on GCP Bucket

Now I will store the output file on GCP bucket storage. The user also can download the output csv file from GCP bucket .the following code will upload the output csv file on Bucket :

Uploading the output file on GCP Bucket

```

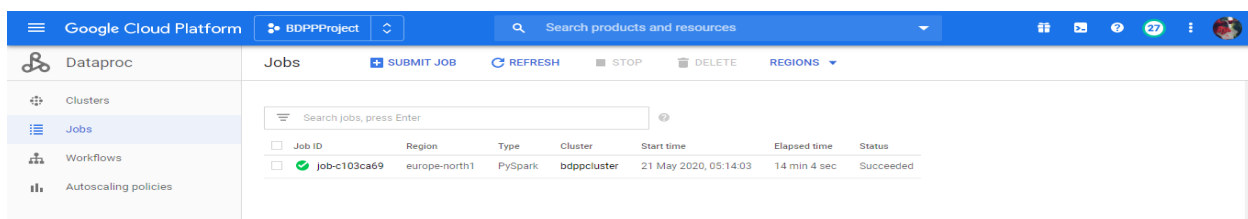
#Uploading the file to cloud bucket
import random
random_num=random.randint(0,20)
file_path="gs://ajmal1/output" +str(random_num)

df=best_predictions.toPandas()
csv = df.to_csv(file_path+".csv")
print("your download file name and url", file_path)

```

3.9 Job submission

after completing the code in jupyter notebook now I must submit a cluster job with main python file and cluster will run the job on GCP we can see all the output on the GCP as well .the following screenshot containing the job status, job id and other credential :



Google Cloud Platform						
BDPPProject						
Search products and resources						
Dataproc						
Jobs						
SUBMIT JOB REFRESH STOP DELETE REGIONS						
Search jobs, press Enter						
Job ID	Region	Type	Cluster	Start time	Elapsed time	Status
job-c103ca69	eu-west-1	PySpark	bdppcluster	21 May 2020, 05:14:03	14 min 4 sec	Succeeded

After submitting the job, we can see the job detail which is containing the job output configuration etc. You can see the following screen shot:

Google Cloud Platform | BDPPPProject | Search products and resources

Dataproc | Job details | REFRESH | CLONE

job-c103ca69
Start time: 21 May 2020, 05:14:03 | Elapsed time: 14 min 4 sec | Status: Succeeded

Output | Configuration

☐ Line wrapping | Equivalent command line

```
LVC Hyper ROC 0.8628705536803694
[[{"LogisticRegressionModel": {"uid": "LogisticRegression_a9a6bb683842", "numClasses": 2, "numFeatures": 13, "0.87883684270021373}, {"CrossValidatorModel": {"uid": "CrossValidatorModel1_8a7da4a49295", "0.87337595104841853}, {"
All Model with Accuracy [{"DecisionTreeClassifierModel": {"uid": "DecisionTreeClassifier_7151dea45ca0", "depth": 3, "numNodes": 11, "0.7889001029973923}, {"CrossValidatorModel": {"uid": "CrossValidatorModel1_ab21c97c439",
the best model is [{"CrossValidatorModel": {"uid": "CrossValidatorModel1_d9cf700d6984", "0.9064983103409463}]]
+-----+
|age|label|rawPrediction|prediction|
+-----+
| 28 | 0.0 | [0.90374032467220... | 0.0 |
| 33 | 1.0 | [-0.5203579257339... | 1.0 |
| 33 | 0.0 | [1.01274938190622... | 0.0 |
| 36 | 1.0 | [-0.6218952477054... | 1.0 |
| 40 | 0.0 | [-0.4940102041140... | 1.0 |
| 42 | 0.0 | [-0.1394899229710... | 1.0 |
| 42 | 1.0 | [-0.4919674767176... | 1.0 |
| 45 | 0.0 | [0.00481205929246... | 0.0 |
| 56 | 0.0 | [0.01623950985677... | 0.0 |
| 20 | 0.0 | [1.16328856057226... | 0.0 |
+-----+
only showing top 10 rows

ROC 0.9064983103409462
your download file name and url gs://ajmal1/output11
20/05/21 03:28:00 INFO org.spark_project.jetty.server.AbstractConnector: Stopped Spark@222475df(HTTP/1.1,[http/1.1])(0.0.0.0:0)
Job output is complete
```

Activate Windows
Go to Settings to activate Windows.

After this we can see our Bucket that has output csv file on GCP bucket as well . Following screenshot containing the bucket file detail

Google Cloud Platform | BDPPPProject | Search products and resources

Storage | Bucket details | EDIT BUCKET | REFRESH BUCKET

ajmal1

Objects | Overview | Permissions | Bucket Lock

Upload files | Upload folder | Create folder | Manage holds | Delete

Filter by prefix...

Buckets / ajmal1

<input type="checkbox"/>	Name	Size	Type	Storage class	Last modified	Public access	Encryption	Retention expiry date	Holds	
<input type="checkbox"/>	bdpp_final_proj.py	20.31 KB	text/plain	Standard	21/05/2020, 04:20:35 UTC+2	Not public	Google-managed key	-	None	
<input type="checkbox"/>	data_frame.csv	5.08 MB	application/vnd.ms-excel	Standard	13/05/2020, 05:31:52 UTC+2	Not public	Google-managed key	-	None	
<input type="checkbox"/>	output11.csv	7.29 MB	application/octet-stream	Standard	21/05/2020, 05:27:51 UTC+2	Not public	Google-managed key	-	None	

4 Problems and solution on GCP

Problem1: pyspark did not load directly from the GCP bucket and topandas() function is not working

- ✓ Create d a cluster with spark 2.4.5 slove the problem

Problem2: Matplotlib and google.cloud libraries are not working on jupyter notebook

- ✓ Install explicitly libraries on master node with pip command

Problem:3 creating a firewall rule with ip range 0.0.0.0

- ✓ Solve it by giving the ip with subnet mask also like: 0.0.0.0/0

Problem4: Job submission with ipynb file

- ✓ Solve it by changing the file type ipynb to .py

5 Conclusion

Finally, I have done all the step with code and GCP task. I have done five different algorithms with pyspark and got the Roc of each model. Then I select the best algorithms Gradient Boost Tree Classification for census income dataset with 0.90 ROC .The Roc is depend upon the dataset that how we prepare the data for the machine learning .after this I used GCP for hosting the project as the GCP has more features than IBM Watson. GCP is realy big platform and has bigquery , storage , network , firewall and much others interesting features. The future scope of this work involves achieving an over-all better set of results by using hybrid models with inclusion of Machine Learning and Deep Learning together, or by applying many other advanced preprocessing techniques without further depletion in the accuracy.