**Solution Abstract : CCP Data Scientist Fall 2014**

**Problem 1: SmartFly**

**Approach:**

**a) Data preprocessing:**

1. step 1: Remove all those data from training data when flight has been cancelled. We are not taking those flight information which has been cancelled.
2. step2 : Replace all Null value with zero anywhere in training data.

**b) Feature Selection:** Intuitively, most influential attributes to catch the seasonality of data would

be to use Month, Day of month, day of week. Total flying time duration is seems to importance

along with Airline , Flight number, Origin airport attribute, Plane model , Tail number as

categorical attribute to catch specific information related to weather and geographical condition.

But after careful investigation we came to conclusion that among them attribute Plane model is

most influential attribute along with Airline attribute.

**Attributes used to train model are :**

1. Month(attr3)
2. Day of month(attr4)
3. Day of week(attr5)
4. Scheduled departure time(attr6)
5. Scheduled arrival time(attr7)
6. Airline(attr8)
7. Plane model(attr11)
8. Distance travelled(attr16)
9. target label : target

After all preprocessing and feature selection data is stored in training\_data.csv file.

**Assumptions:** The positive time delay in departure time is considered delayed flight else no delay.

To label the class(delay or no delay), we use 1 for delay and 0 for no delay.

**Software Tools used :** R programming with RStudio with gbm R package

**Algorithms:** The best model performance we got with gradient boosting model using gbm r package

in rstudio.

**Code To Run GBM Model :**

Code to run gbm model(with all parameter value)

gbm.fit(x, y,

offset = NULL,

misc = NULL,

distribution = "bernoulli",

w = NULL,

var.monotone = NULL,

n.trees = 100,

interaction.depth = 1,

n.minobsinnode = 10,

shrinkage = 0.001,

bag.fraction = 0.5,

nTrain = NULL,

train.fraction = NULL,

keep.data = TRUE,

verbose = TRUE,

var.names = NULL,

response.name = "y",

group = NULL

)

**Testing and validation techniques applied**: 10 cross validation

**Model selection criteria**: We split the data into 9/10 and 1/10 part. 9/10 data is being used to train

the model and other 1/10 data to test the performance of model. We used ROC to measure the

performance of the model. since we had to predict the probability of delay in flight and ROC

metric measure area under the curve for probability and class labels.

**PS:** train\_data.csv is training data used to train the model. This file is generated after

preprocessing of smartfly\_historic.csv data. problem1.csv is output for this problem.

flight\_rank.r is the r code which has been used for this problem.

**Part 2: Almost Famous**

**Problem solving Approach and Methodology:**

Using the right file format is critical for getting performance while using Impala , since spam.log and the web.log file are given in the json format this needs to be converted into AVRO file format first , which ensures Impala queries can be run with good efficiency.

**Data Preprocessing:**

**Step1:** Is to convert spam.log file into Avro format with the help of Pigscript

REGISTER piggybank.jar

data = load '/user/hive/warehouse/almostfamous/spam.json'

AS (visit\_id:INT, uid:INT, campaign:INT, tstamp:CHARARRAY, experiments:INTARRAY, action:CHARARRAY, query:CHARARRAY);

data\_clean = FILTER data BY uid IS NOT NULL AND visit\_id IS NOT NULL;

STORE data\_clean INTO 'impala/spam\_avro'

USING org.apache.pig.piggybank.storage.avro.AvroStorage(

'{

"schema": {

"name": "spamavro",

"type": "record",

"fields": [

{"name":"visit\_id", "type":"int"},

{"name":"uid", "type":"int"},

{"name":"campaign", "type":"int"},

{"name":"tstamp", "type":"string"}

{"name":"experiments", "type":"record","fields":[{"name":"Exp\_A","type":"int"},{"name":"Exp\_B","type":"int"}]}},

{"name":"action", "type":"string"},

{"name":"query", "type":"string"},

]}

}');

**Step2:**  Use the Hive Editor to create the table for both spam.log and web.log as spam\_avro, web\_avro respectively with the below schema

CREATE TABLE spam\_avro

ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'

STORED AS

inputformat 'org.apache.hadoop.hive.ql.io.avro.AvroContainerInputFormat'

outputformat 'org.apache.hadoop.hive.ql.io.avro.AvroContainerOutputFormat'

LOCATION '/user/test/impala/spam\_avro'

tblproperties ('avro.schema.literal'='{

"name": "spamavro",

"type": "record",

"fields": [

{"name":"visit\_id", "type":"int"},

{"name":"uid", "type":"int"},

{"name":"campaign", "type":"int"},

{"name":"tstamp", "type":"string"}

{"name":"experiments", "type":"record","fields":[{"name":"Exp\_A","type":"int"},{"name":"Exp\_B","type":"int"}]}},

{"name":"action", "type":"string"},

{"name":"query", "type":"string"}]}'

);

**Step3 :** login to impala and REFRESH spamlog\_table, weblog\_table created

**Software Tools Used :** Apache Pig Script, Impala, Hive Table Creation in HDFS

**Total time spent:** 36 Hours

**a)** counting number\_of\_distinct\_visitors\_that\_are\_bots

data = LOAD 'spam.log' using PigStorage(',');

data0 = FOREACH data GENERATE $0 AS key:chararray, $1 AS key2:chararray;

data1 = distinct data0;

dump data1;

user\_list = foreach data1 GENERATE $1;

unique\_bot\_users = DISTINCT user\_list;

**b)** overall clickthrough rate of the ads

**Part 3: WINKLR**

**Approach:**

**a) Data preprocessing:**

1. step 1: Convert the data frames into graphs, The testing and the training data can be converted to SNA, social network analysis compatible graphs .
2. step2 : Compute the matrix of adjacency usin g igraph,sna( social network analysis module present in the R package in R Studio.

**b) Feature Selection:** The SNN was used since it considers not only direct associations between vertices but also indirect connections. This characteristics of SNN can be used to find similarities between vertices that are not adjacent also. The accuracy and performance of the recommendation system was compared by applying the Neumann Kernels which considers the concept of proximity between the edges of the graph.It was found that both the algorithms were effective to predict the “user1,user2” tuples where user1 is mostly likely want to follow user2.

**Attributes used to train model are :**

The tuples provided in the data set are used as data input to the recommender system.

**Software Tools used:** R programming with RStudio with igraph, sna (social network analysis) R package

**Algorithms:** The SNN, Neumann Kernel algorithms are used to solve the problem.

**Total Time Spent :** 48 hours

**Solution:** The tuples who are likely to follow each other are placed in the problem3.csv file