CS6240
PARALLEL DATA PROCESSING WITH MAP-REDUCE
FINAL PROJECT

BY,
SANJANA MANOJ KUMAR
SUNDEEP ANNA LOGANATHAN

MODEL USED AND PARAMETERS EXPLORED:

FINAL APPROACH:
Prediction model used: LogitBoost
Accuracy: 76%

INITIAL APPROACH:

Initially, we attempted to use **Random Forests** as our prediction model. We were able to achieve this and generate predictions with an evaluated accuracy of **83-84%** when the training and testing took place in the same map reduce job.
After cleaning the labelled and unlabelled data, the next map reduce job used multiple input mapper to send training data in random to each reducer and the test data was sent to all the mappers.
In the reducer, the models were first trained and evaluated on the train data and once the mode was built, the predictions for the test data was made in the same reducer. I.e the models were not persisted.

MODIFICATION TO INITIAL APPROACH:

Since 2 syslog files were part of the requirement, we attempted to change our initial approach to 2 map reduce programs. The output for the first program would be the models and the second program would take the models and unlabeled dataset as input. The **difficulty** with this approach was that once the modes were read back to memory, it would throw errors for the same instances as our initial approach.

FINAL APPROACH:

This led us to our final approach. On further exploration, we realized that this problem existed with any model that used a tree structure and finally opted for Logistic regression.

VALIDATION:

Weka has inbuilt functionality to cross validate models. We used this to gauge accuracy and tune the parameters.

PRE-PROCESSING:

FIELDS CONSIDERED:

Total number of fields considered: 17 and the label (Agelaius_phoeniceus)

LATITUDE, LONGITUDE : Location is a very important factor to determine presence/absence of a bird
YEAR, MONTH, DAY, TIME: Birds follow very specific patterns and chances of spotting them at a particular time is high

NUMBER_OBSERVERS: Number of people that observed the bird. Could be a valid field incase someone missed the bird
POP00_SQMI: Human population
HOUSING_DENSITY, HOUSING_PERCENT_VACANT : Urbanization can be a factor for the bird's appearance/ disappearance
ELEV_GT, ELEV_NED: Elevation above ground level
BCR: Birding region
CAUS_TEMP_AVG, CAUS_TEMP_MIN, CAUS_TEMP_MAX, CAUS_PREC: Temperature (weather conditions)
We tried to keep the number of fields as small as possible so that the importance of each field is not lost i.e., adding more features can add noise by taking away importance from the important ones.
All the fields considered are from the Checklist or Core covariates. We also made sure to take continuous variables since we are opting for Logistic regression.

The 'Agelaius_phoeniceus' field had some '?' and 'X' present. We replaced the '?' with 0 and the 'X' with 1 because 'X' represents that the observer has lost track of the number of times he spotted the bird.
Any other missing data was replaced by 0.

PSEUDOCODE

MODEL TRAINING PROGRAM:

The model training program consists of two map-reduce jobs:

1. Cleaning labeled data
2. Creating models

CLEANING LABELLED DATA:

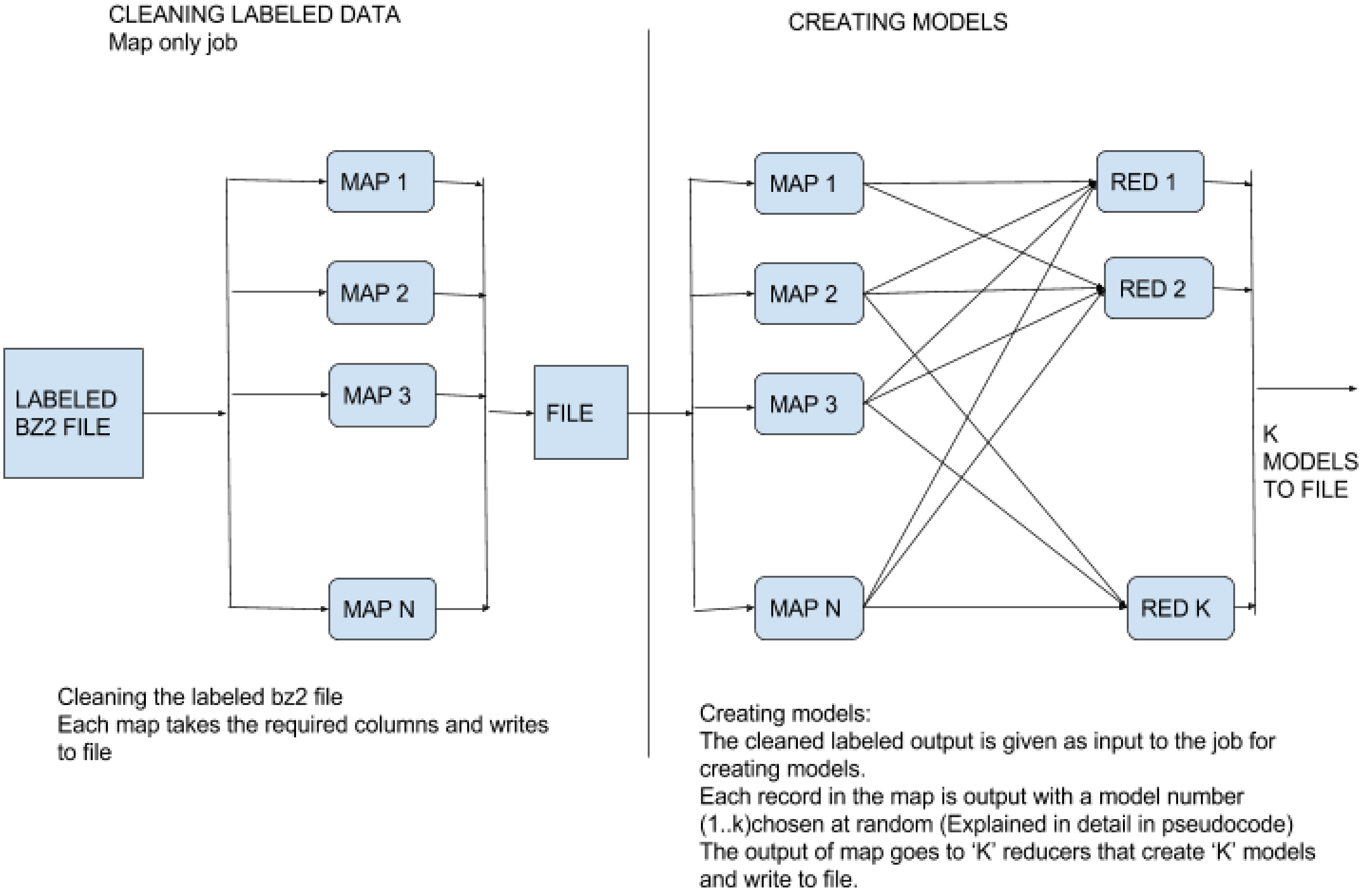
It is a map only job.
The cleaned data is written to file and is fed as input to the model training job.

MAPPER

```
{
        map(Object obj, Text line)
        {
                //for each line take only the required columns and append          to data, separated by ','
                String data = required columns for prediction
                data+=column27 // the bird field
                cleanedData = Replace the missing data ('?', 'NA'. 'X')
                emit(cleanedData, null)
        }

}
```

DIAGRAMMATIC REPRESENTATION:



CLEANING LABELED DATA
Map only job

CREATING MODELS

Cleaning the labeled bz2 file
Each map takes the required columns and writes to file

Creating models:
The cleaned labeled output is given as input to the job for creating models.
Each record in the map is output with a model number (1..k)chosen at random (Explained in detail in pseudocode)
The output of map goes to 'K' reducers that create 'K' models and write to file.

TRAINING THE MODEL:

The labelled data is very biased and the ratio between the samples where the bird is present to not present is huge. Therefore, we make sure that every sample which has the bird spotted is included for training and the samples which do not have the bird are randomly sampled. This is taken care of in the Mapper. This is how bagging is performed.

MAPPER

```
{
  Random r;
  int k;
  setup()
```

```
   {
          //instantiate r
          //k is the number of models that is loaded from the configuration
   }

   map(..,training record r)
   {
          String[] fields = split r by','
          if(fields[fields.length] >0) //i.e the bird is present
                  emit(rand%k, r)
          else
                  if(rand%17<8)
                          emit(rand%k, r)
   }
}
```

All the training records for each model come to a different reducer (Multiple keys can go to the same reduce task. I mean different reduce calls in this scenario) . Weka instances are created the model is trained and evaluated in the reducer before writing the models to file.

REDUCER

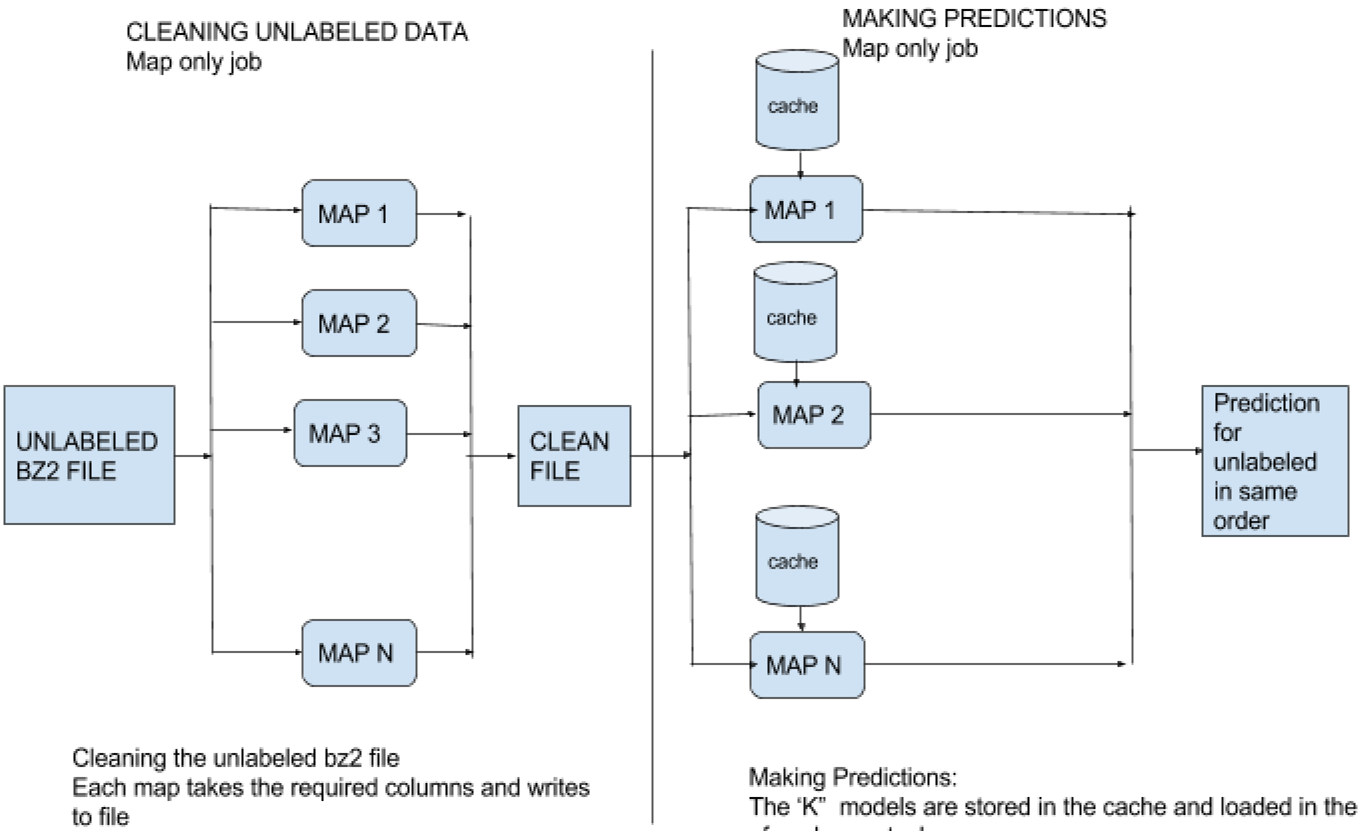
```
{
   reduce(key modelNum, Iterable<Text> trainingRecords)
   {
          newData //Intances of all the training records for the model
          //Split newData to training and testing set
          Classifier model = new Logistic() // Creates new logistic regression classifier
          model.buildClassifier(newData)        // builds the classifier based on the newData
          //Evaluate the model and print accuracy
          //Write the model to file
   }
}
```

PREDICTION PROGRAM:

It consists of two parts:

1. Cleaning unlabeled data
2. Predicting whether the bird is present

DIAGRAMMATIC REPRESENTATION:



Cleaning the unlabeled bz2 file
Each map takes the required columns and writes to file
Order is maintained since it is a map only job

Making Predictions:
The 'K" models are stored in the cache and loaded in the setup of each map task
The cleaned unlabeled output is given as input to the job for making predictions for each record for each model i.e., each record 'r' in the cleaned file gets 'K' predictions.
The predictions are aggregated and based on a threshold, choice is made.
Again order is maintained since it is a map only job.

CLEANING UNLABELED DATA:

The cleaning of unlabeled dataset is similar to that of the labeled dataset. We included the samplingID to the end of the records as it is required in the output. This is a map only job and thus the input order is maintained at the output.

MAPPER

```
{
        map(Object obj, Text line)
        {
                //for each line take only the required columns and append  to data, separated by ','
                String data = required columns for prediction
                data+='?' // the bird field is replaced by a ?
                data+=column0 // the sampling ID
                emit(data, null)
        }

}
```

This is a map only job. The input to this job is the cleaned unlabeled data. The map reads the 'K' models and in the setup from cache and each input record in predicted by each model. The total is aggregated and the average is calculated. Based on the average the prediction result is decided. The output maintains same order as input since it is a map only job.

PREDICTING WHETHER THE BIRD IS PRESENT:

MAPPER

```
{
  setup()
  {
        models = read all the models from file cache
  }

  map()
  {
        for each M in Models do
                compute M(t) and update the sum and count
        if(sum/count)>=0.5
                emit(t,1)
        else
                emit(t,0)
  }
}
```