Bird Sighting Predictions

Data Mining with Big Data

Problem Tackled

Implemented a Distributed Parallel Algorithm to predict the presence or absence of the Red-winged Blackbird (Agelaius phoeniceus) in each birding session as accurately as possible.

The Project uses Hadoop MapReduce framework with Weka as the library for Data Mining. It implements several algorithms and design patterns learnt in CS6240 under the guidance of Prof. Nat Tuck.

Key Points

1. Labelled Data is split into 80-20 for building a model and validation, with only the required attributes.
2. Ensemble algorithm is used with building a configurable number of models of Random Trees and the final prediction is achieved via Voting.
3. Each Random Tree is built in parallel on separate worker machines with tuned parameters which are set according to the suggestions made in recent studies and experimental findings.
4. Predictions are done in parallel and is scalable to the number of records to be predicted.

Approach

3 MapReduce Jobs are implemented that achieve the following tasks :

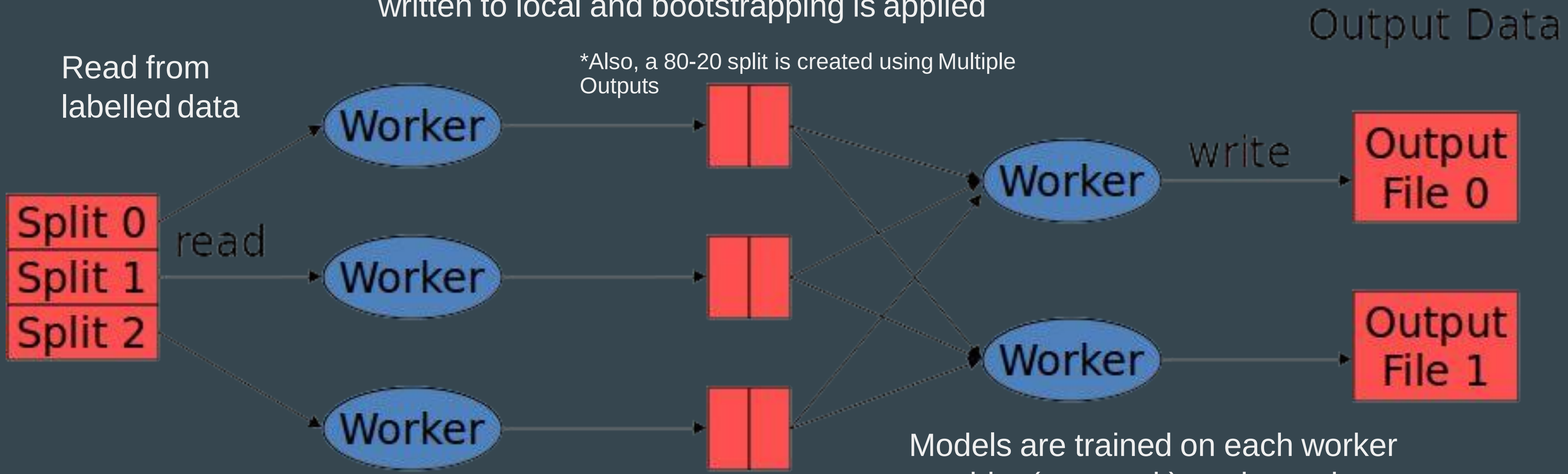
1. Performing preprocessing on the labelled data - Job 1
2. Building the model - Job 1
3. Validation of the model and finding the accuracy - Job 2
4. Predictions - Job 3

Job 1-Preprocessing and Modelling

Formatted data with key as the model number is written to local and bootstrapping is applied

Read from labelled data

*Also, a 80-20 split is created using Multiple Outputs



Models are trained on each worker machine(map task) - using weka

Partitioner is used here to ensure that load is distributed amongst the reducers

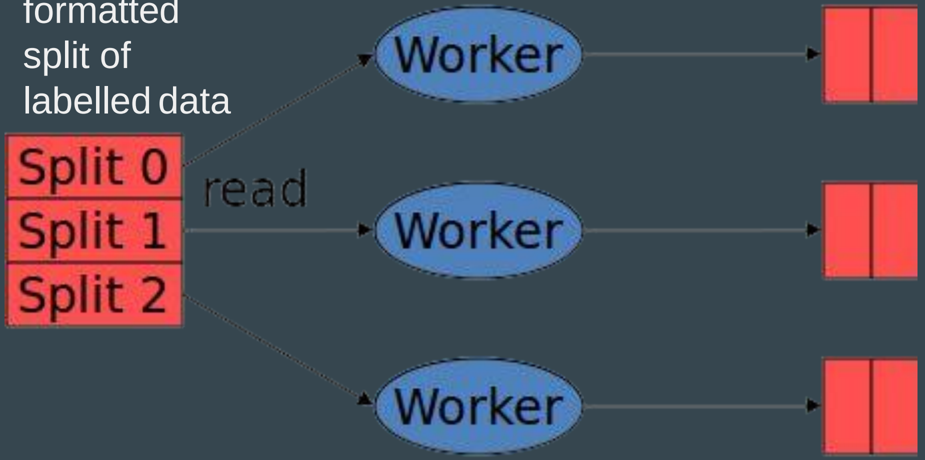
Job 1-Preprocessing and Modelling -Features

- Random Shuffling
- Preprocessing , data split and modelling in a single job
- Partitioner used for load distribution
- Attributes selected for Modelling :
  - Latitude and Longitude
  - Date , Time , Month and Year
  - Data related to housing density at the sampling site
  - Data related to elevations at the sampling site
  - Data related to water bodies near or at the sampling site
  - Also , data was filtered to get distinct records and remove redundant observations from an observing group members

Job 2 - Validation

Read from the 20% formatted split of labelled data

All the models are read from the disk and loaded in memory, once for each map task



Map Only Job which uses the models loaded in memory and make final predictions using the majority of predictions made from amongst all the models.

Predictions are made for each map call and is sent to output.

Global counters are kept to keep a count of correct and incorrect predictions based on the actual value in the record sent to mapper

Job 2 - Validation - Features

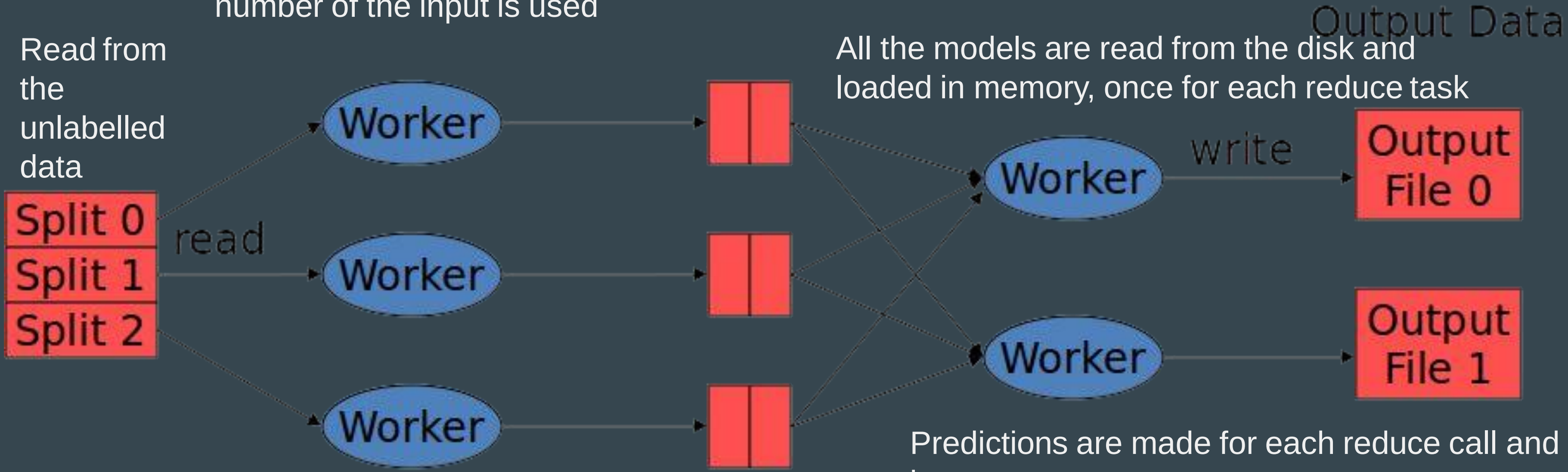
- Minimum I/O - Map only Job
- Hadoop counters used for calculating Accuracy of the model

Job 3 - Predictions

Formatted data with key as the indexed line number of the input is used

Read from the unlabelled data

All the models are read from the disk and loaded in memory, once for each reduce task



Predictions are made for each reduce call and is sent to output.
Final prediction is made using the majority

Partitioner is used here to ensure that load is distributed amongst the reducers and to ensure total order partitioning.

Job 3 - Predictions

- Scalable solution - can use up to “n” machines

Design Decisions

- Why forest of Random Trees?

Random Trees proves to be the right choice for unstable and skewed Data like ours. Using a forest of Random Trees made us exploit the distributed parallel feature of MapReduce to a good extent to train models parallely on worker machines.

Also , the accuracy we obtained using forest of Random trees with Bootstrapping/Bagging was the highest out of all the models weexplored.

The approach achieves a lower test error solely by variance reduction. Since each random tree is trained on random subset of bagged data and random attributes out of all the attributes.

Design Decisions

- Deciding ‘N’ for number of models?

After trying with lot of values for ‘N’ we selected 10 as the Results was efficient and Prediction were reaching an accuracy of 83%.

- Parameters explored for Random Trees?

After researching and experimenting , we found the following parameter to be best suitable for Random trees :

Minimum size of the each node of a tree = 1,
Attributes to randomly select for building the tree= square root of total attributes(25) ,
Depth of Tree= 0(Infinity)

Accuracy RoadMaps

- Single NaiveBayes Weka Model :
  - 67%
- Single Predefined Mboost Weka Pre-Defined:
  - 74%
- Single Random Tree on 1/10th:
  - 75%
- n Random Trees on 1/n of 80% Training Data:
  - 81%(n=10) 78%(n=100)
- 10 Random Trees with Bagging/Bootstrapping of 80% Training Data with Bagging
  - :83.7%