A7: Routing

Harshali Singh, Vishal Mehta

- Strategy
- Study of Performance
- Breakdown of Work
- Conclusion

Strategy

1. Modeling: We have used weka API to build our predictive model. In that we first express the problem with features, train a classifier and then use that classifer to set the dataset and produce the output.

    i. Expressing problem with features(Attributes): We have 8 numeric features and 1 nominal feature (ARR_DELAY) in our program.

    ```
        attributes.add(new Attribute("dayOfMonth"));
        attributes.add(new Attribute("dayOfWeek"));
        attributes.add(new Attribute("carrier"));
    ```

    ii. Train a classifier: Training requires 1) having a training set of instances and 2) choosing a classifier(Naive Bayes here).

    ```
    iFlight.setValue((Attribute)fvWekaAttributes.get(0), flight.getDayOfMon
    th());
    iFlight.setValue((Attribute)fvWekaAttributes.get(1), flight.getDayOfWee
    k());
    iFlight.setValue((Attribute)fvWekaAttributes.get(2), flight.getCarrier
    ().hashCode());
    ```

    iii. Use the classifier.

    ```
                iFlight.setDataset(isTestingSet);
                double[] predictionDistribution = classifier.distributionFo
    rInstance(iFlight);
                String prediction = null;
                if(predictionDistribution[0] > predictionDistribution[1]){

                    prediction = "TRUE";

                } else prediction = "FALSE";
    ```

2. Prediction

We have used Naive Bayes theorem to predict the delays. Naive Bayes is a simple probabilistic classifier based on applying Bayes' theorem (or Bayes's rule) with strong independence (naive) assumptions.Bayes's rule: P(H | E) = P(E | H) x P(H) / P(E) The basic idea of Bayes's rule is that the outcome of a hypothesis or an event (H) can be predicted basedon some evidences (E) that can be observed.

In our program, we have taken DayOfMonth, DayOfWeek, Carriercode, Origin , Destination, Sch dep Time, Days Till Nearest Federal Holiday, Arrival delay as numeric features while training the classifier.

Here, we have assumed that if the flight is delayed (ARR_DELAY > 0) then it is considered as MISSED.

3. FLOW OF THE PROGRAM:

The Routing Mapper gives the output having month as a key and all the other attributes as value to the reducer. The Routing Reducer train the models(based on month) and use the classifier to emit the data based on monthly models using Naive Bayes classifier of Weka.

After that, the program reads the test file, uses the monthly models to predict if the flight is missed or not using probability prediction distribution as described above.

In the Connection Mapper, we are filtering the records that do not pass the sanity test. We have created two Key-Value pairs for each flight.In the first pair we consider the arrival information of the flight and in the second pair we consider the departure information of the flight.

F Data(Arriving Flight) :-

Key: Year, Month, Carrier,Destination Airport Code Value: Arriving Flight's FlightWritable object

G Data(Departing Flight) :-

Key: Year, Month, Carrier, Origin Airport Code Value: Departing flight's FlightWritable object

The output of the Routing job is fed into the Connection job.

In Connection Reducer, we iterate over all the values and identify that the value is F data or G data. However, we only classify F data as F describes the arriving flight using the Naive Bayes Classifier.

The final output from Connection Reducer would be connections and missed connections as the values in respective outstream. The key is Year, month, dayOfMonth, F origin Airport, G destination airport. The value is F Flight number, G Flight number, duration.

Again the output of the Connection job is fed into the Request job.

In the RequestMapper we read the request file and Connection output to emit the pair in which Key is the same as of Connection's output key and value is the same as of Connection's output value. This pair is passed to Reducer to find out the optimal flight routes.

4. CONFUSION MATRIX

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.

Terminologies:

```
a. true positives (TP): These are cases in which we predicted yes (flight is misse
d), and the flight actually got missed.
b. true negatives (TN): We predicted no, and the flight is not missed.
c. false positives (FP): We predicted yes, but the flight is not missed.
d. false negatives (FN): We predicted no, but the flight is actually missed.
```

We have written a Spark program to generate the confusion matrix which takes the output of the Prediction program and a validate file to determine the accuracy of the prediction. The accuracy of the program from confusion matrix comes out to be : 49.043752167% approx. (TP+TN/TP+TN+FP+FN)

Study of Performance

We have used python script to ping the emr cluster to know whether it has terminated or not. It also calculates the running time of the program.

Routing on pseudo: 12 mins approx Routing on EMR: 8 min approx

Breakdown of Work

Vishal Mehta: Worked on building the code that takes request file and generate the optimal flight routes, prepared this report and wrote the code to generate the confusion matrix.
Harshali Singh: Worked on building the model and Connection code that takes test file and emits the connections and missed connections in two different files. Also contributed in preparing this report. We both ran the code in EMR and pseudo.

Conclusion

Based on the accuracy of the solution above, it shows that the predicted optimal flight routes are 50% accurate against the given validated flight routes. We could have also tried altering the predictive model to include another nominal attribute - isMissed?.We could have used the Missed Connection's assignment to train the data whether the connection was missed or not.