Assignment 2 report

Q1

1.

a. Use randomSplit to get random 1% of the data.

b. The best parameters are found using grid search as shown in the code
For random forest model, the parameters lists:

```
paramGrid = ParamGridBuilder() \
    .addGrid(rf.featureSubsetStrategy, ['all','sqrt','log2']) \
    .addGrid(rf.maxDepth, [1, 10, 15]) \
    .addGrid(rf.numTrees, [1, 10, 20]) \
    .build()
```

The best list are: featureSubsetStrategy = ‘all’, maxDepth = 10 and the numTrees = 20.

For GBT model, the parameters lists:

```
paramGrid = ParamGridBuilder() \
    .addGrid(gbt.maxDepth, [1,10,15]) \
    .addGrid(gbt.maxIter, [10, 20, 30]) \
    .addGrid(gbt.featureSubsetStrategy,['all','sqrt','log2']) \
    .build()
```

The best list are: featureSubsetStrategy = ‘all’, maxDepth = 10 and the maxIter = 30.

For NN model, the parameters lists:

```
paramGrid = ParamGridBuilder() \
    .addGrid(mpc.maxIter,[30,50,100]) \
    .addGrid(mpc.blockSize,[64,128,256]) \
    .addGrid(mpc.stepSize,[0.01,0.03,0.05]) \
    .build()
```

The best list are: featureSubsetStrategy = ‘all’, stepSize = 0.01 and the maxIter = 100.

c. Comparing the performance:

Accuracy for best random forest model on small data set = 0.709
Auc for best random forest model on small data set = 0.796

Accuracy for best GBT model on small data set = 0.704
Auc for best GBT model on small data set = 0.795

Accuracy for best MPC model on small data = 0.683
Auc for best MPC model on small data = 0.758

2. Working with the larger dataset.

a. As shown in the code, the best parameters were used for each model on larger dataset.

b. Comparing the three different methods:
Accuracy for best random forest model on larger data set = 0.707
Auc for best random forest model on larger data set = 0.796

Accuracy for best GBT model on larger data set = 0.733
Auc for best GBT model on larger data set = 0.795

Accuracy for best MPC model on larger data = 0.684
Auc for best MPC model on small larger set = 0.758

c. Provide training time when using 10 cores and 5 cores.
The time for random forest with 5 cores is 18.56 min.
The time for random forest with 10 cores is 17.16 min.

The time for Gradient boosting with 5 cores is 23.68 min.
The time for Gradient boosting with 10 cores is 25.1 min.

The time for shallow Neural network with 5 cores is 66.48 min.
The time for shallow Neural network with 10 cores is 66.1 min.
The reason for the long training time for the NN model is that the number of layers is large and the step size is small, which makes it easier to get good performance but also takes more time. With maxIter = 30 and stepsize = 0.1, it takes about ten minutes, but the accuracy drops to about 0.6.

3. The three most relevant features for ensemble methods：

Top 3 for random forest: _c26, _c28, _c27
Top 3 for Gradient boosting: _c26, _c28, _c25

4. Observations

1. An increase in core count will not necessarily save training time.
2. Using different metrics (accuracy and auc) to assess the predictions will get different results.
3. Different models give different feature importance.
4. The number of cores used for training does not affect the performance of the model.

Q2

1. Preprocessing

a. The percentage of missing values in each column is counted, and the 4 columns (Cat2, Cat4, Cat5, Cat7) with the worst missing values are dropped (these 4 columns have 30% to 50% missing values). For the remaining missing values, drop the corresponding rows directly. This way you can get data without missing values while keeping most of the data. Columns (Row*ID, HouseHold*ID, Blind*Submode, Calendar*Year, Blind_Make) that are not relevant to the predicted outcome are removed as well.

b. As shown in the code, data of type String is found and processed using StringIndexer to convert the string type data into the appropriate form the store them in the new column with column name like X_num.

C. As shown in the code, oversampling is used to balance the data, as the data is extremely unbalanced and a lot of data would be lost if downsampling was used. After the oversampling was used the claim/unclaim is 0.996.

2. Prediction using linear regression.

a. The mean absolute error on test data: 94.55.
The mean squared error on test data: 10259.78

b. Total time of LinearRegression in 5 cores condition: 2.38 minutes.
Total time of LinearRegression in 10 cores condition: 1.23 minutes.

3. Prediction using a combination of two models.

a. First model is LogisticRegression model which is used to class 0 claim and null 0 claim.

b. The second model is GLR model which is based on the prediction of the result of first model. The mean absolute error on test data is 1.38 and the mean squared error is 1586.26

c. Total time of combined model in 5 cores condition: 19.15 minutes. Total time of combined model in 10 cores condition: 9.78 minutes.

4. Observations

1. Training with a 10 core will be about twice as fast as training with a 5 core.
2. A single linear regression is faster than a combined model.
3. A joint model works better than a single model.