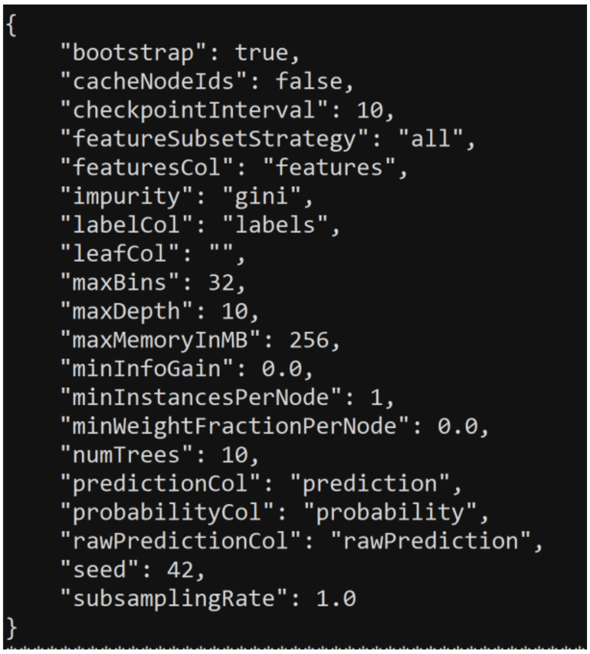
COM6012 Assignment 2

Question 1. Searching for exotic particles in high-energy physics using classic supervised learning algorithms

1. Use pipelines and cross-validation to find the best configuration of parameters for each model.
    a. We first randomly chose a small subset of data (10%) by using the sample function to find the best parameters for each of the models.
    b. For Random Forrest Classification we have used the following parameters: maxDepth, numTrees and featureSubsetStrategy. For Gradient Boosting Classification we have used maxDepth, maxIter and maxBins. For Shallow Neural Network we have used layers, maxIter and stepSize.
    c. To have the same train and test data constant throughout we have used the parquet function and saved the copy on the drive.

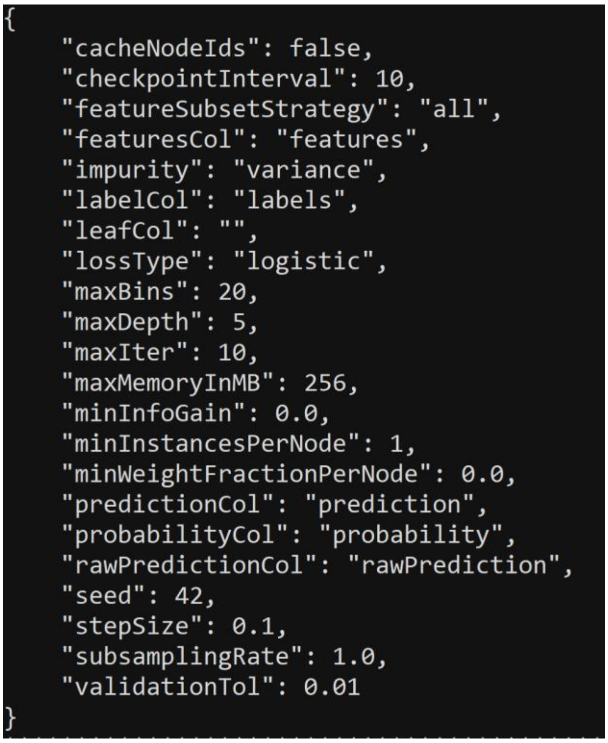
Model	AUC	Accuracy
Random Forrest	0.7067	0.7082
Gradient Boosting	0.6950	0.6957
Shallow Neural Network	0.6332	0.6506

2. Working with the larger dataset: We have applied the randomSplit function to split the large data into training and test in the ratio 70:30.
    a. Best Parameters:
    Random Forrest:



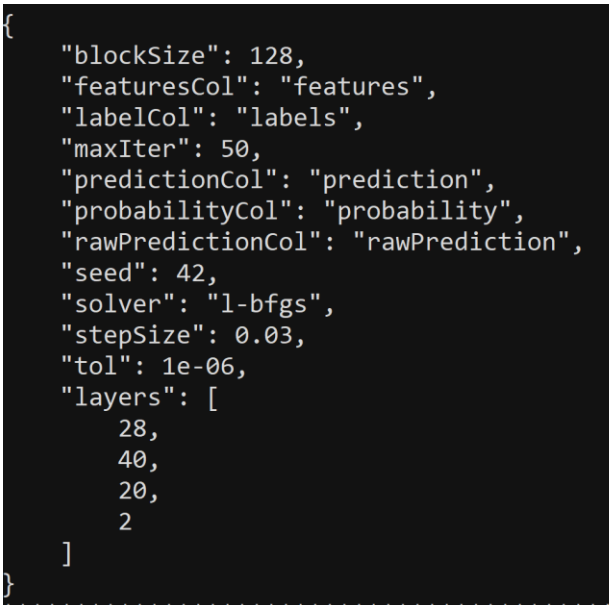
```
{
    "bootstrap": true,
    "cacheNodeIds": false,
    "checkpointInterval": 10,
    "featureSubsetStrategy": "all",
    "featuresCol": "features",
    "impurity": "gini",
    "labelCol": "labels",
    "leafCol": "",
    "maxBins": 32,
    "maxDepth": 10,
    "maxMemoryInMB": 256,
    "minInfoGain": 0.0,
    "minInstancesPerNode": 1,
    "minWeightFractionPerNode": 0.0,
    "numTrees": 10,
    "predictionCol": "prediction",
    "probabilityCol": "probability",
    "rawPredictionCol": "rawPrediction",
    "seed": 42,
    "subsamplingRate": 1.0
}
```

Gradient Boosting:



```
{
    "cacheNodeIds": false,
    "checkpointInterval": 10,
    "featureSubsetStrategy": "all",
    "featuresCol": "features",
    "impurity": "variance",
    "labelCol": "labels",
    "leafCol": "",
    "lossType": "logistic",
    "maxBins": 20,
    "maxDepth": 5,
    "maxIter": 10,
    "maxMemoryInMB": 256,
    "minInfoGain": 0.0,
    "minInstancesPerNode": 1,
    "minWeightFractionPerNode": 0.0,
    "predictionCol": "prediction",
    "probabilityCol": "probability",
    "rawPredictionCol": "rawPrediction",
    "seed": 42,
    "stepSize": 0.1,
    "subsamplingRate": 1.0,
    "validationTol": 0.01
}
```

Shallow Neural Network:



```
{
    "blockSize": 128,
    "featuresCol": "features",
    "labelCol": "labels",
    "maxIter": 50,
    "predictionCol": "prediction",
    "probabilityCol": "probability",
    "rawPredictionCol": "rawPrediction",
    "seed": 42,
    "solver": "l-bfgs",
    "stepSize": 0.03,
    "tol": 1e-06,
    "layers": [
        28,
        40,
        20,
        2
    ]
}
```

b. The same training and test data have been used for all the models and was saved using the parquet function on the drive.

c. Training time:

Cores	Model	Times(s)
5	Random Forrest	433.74
	Gradient Boosting	871.29
	Shallow Neural Network	4020.16
10	Random Forrest	386.15
	Gradient Boosting	727.57
	Shallow Neural Network	3788.90

Performance:

Model	AUC	Accuracy
Random Forrest	0.7062	0.7044
Gradient Boosting	0.6954	0.6961
Shallow Neural Network	0.6377	0.6438

3. Most relevant features:

Model	Features	Importance
Random Forrest	m_bb	0.407186
	m_wwbb	0.167812
	m_wbb	0.104551
Gradient Boosting	m_bb	0.172359
	m_wwbb	0.15106
	m_jlv	0.148759

4. Observations:
   a. Columns m_bb and m_wwbb have significant importance as they have appeared in both the models as important features in the same order.
   b. Random Forrest and Gradient Boosting Classifier yielded fairly similar results which was better than the neural network.

Question 2. Senior Data Analyst at Intelligent Insurances Co.

1. **Preprocessing :**

   a. Columns Row_ID and Household_ID are dropped as they bear no significance in predicting the claim amount.
   b. Converted columns Vehicle, Calendar_Year and Model_Year to type “double”.
   c. Missing Data: Checked for "?" in the dataframe and replaced it with “Null”. The “Null” values are then replaced by the most frequent value in the respective column in the dataframe.
   d. Categorical values:

- Since there are many categorical columns without any labels many of them are dropped as it is difficult to derive any intuition from them and also that they would take up the feature space.
- To deal with the categorical values we have applied OneHotEncoding after converting the string values to numeric by using StringIndexer.

e. Unbalanced data: To balance the data we have created a new column claim_derived (1 if Claim_Amount>0 else 0). Since the majority of data has Claim_Amount 0 we have sampled the training data so as to have 95% as the negative cases and 5% as positive. This is achieved by using the sampleBy function in the ratio such as to take up almost every positive case and then selecting the negative cases in a proportion that would equate to 95%.

2. Linear Regression:

a. We have used randomSplit of 70-30 ratio between the train and test data. For this regression we have the Mean Squared Error as 1646.54 and Mean Absolute Error as 10.45.

b. Training Time:

Cores	Time (s)
5	85.10
10	51.24

3. Predicting using combination of two models:

a. Binary Classifier: We have chosen Gradient Boosting to be used as our classifier with the following parameters: maxDepth=10, maxIter=50, maxBins=20, stepSize=0.1, validationTol=0.01, subsamplingRate=1.0, seed=42. For this classifier we got the Mean Squared Error as 0.007610 and Mean Absolute Error as 0.007610.

b. Gamma Regressor: To the predictions made by the Gradient Boosting model we select only those rows which the model has classified as a positive case and pass them to the Gamma Regressor model to predict the claim amount. We have chosen the following parameters: family="gamma", link="identity", maxIter=50, regParam=0.1. We got the Mean Squared Error as 29568.66 and Mean Absolute Error as 167.31.

c. Training Time:

Cores	Model	Times(s)
5	GBT Classifier	1166.82
	GLM	1271.40
10	GBT Classifier	707.04
	GLM	725.68

d. Observations:

    i. The performance metric of the Binary Classifier may seem good but in reality since the data is imbalanced and has majority of negative cases i.e. Claim_Amount = 0 it has been able to predict those cases accurately. Thus as a result when the predictions of the Binary Classifier when passed through the Regression model it returns high error.