**CS 6923 Machine Learning Spring 2019**

**Final Project Report**

**Name: ARJAN SINGH NARULA**

**NetID: asn419**

**Name: Harman Singh Chawla**

**NetID: hsc367**

**PART I: Preprocessing**

**1. How does your program handle missing value? And why?**

* We started our project by generating baseline models and evaluating their performance.
* Since our data is quite unique, and hence traditional imputation and techniques didn’t seem to do much good in our case. Furthermore, there were special cases like “weight” which had no observable values.
* Attempting to generate some missing values (treating them as labels and running simple models) didn’t yield any improvements in the result we were getting baseline through baseline model.
* Thus, features will all missing values were dropped straight away. Samples with more than 25 attributes/features missing were set to be dropped (there were no such samples in the training set)
* This stage was important as we wanted to upsample our data further down the line and missing values would have lead to additional noise in the data.

**2.If your program converts numeric features to categorical features, or categorical features to numeric features. Describe how it does it.**

**Ans:**

* We have used **one-hot encoding** technique which convert the **categorical data into the numerical data**.
* Being careful **not to lose information to techniques like “Label Encoding”** for our numeric features, we individually took columns which represent categorical data and converted them into numerical data.
* For instance, a single value in a column and represent it in a numerical value like Yes for 1 and No for 0 and will continue till it convert all the rows of a particular column into single value.

**3. Describe any feature selection, combination or creation, and any feature values combination performed by your program and the reasons for doing so.**

**Ans:**

* 48 attributes gives our feature-space quite high dimensionality. We used multiple techniques to gauge which features would give the most gain if kept while training the model. This was essential to prevent our model from being sensitive to noise and/or overfitting.
* We have used **correlation** as a technique to see which feature is contributing the most to the final output.
* We have used **correlation matrix**, correlation value and threshold to find out the best features in a dataset. Using this method can help us to identify and remove unneeded, irrelevant and redundant attributes from data that do not contribute to the accuracy of a predictive model or may in fact decrease the accuracy of the model.
* Correlation is not a stand alone measure of best possible features. We used **SelectKBest with chi scoring function** and **ExtraTreesClassifier** to analyse the **features with the most importance**. A union of these features was selected and then thresholded w.r.t their respective correlations.

**4. Describe other preprocessing used in your program(e.g. centralizing, normalization)**

**Ans:**

* It was obvious that our training data was imbalanced. This called for some extensive preprocessing of data in order to ensure that our models weren’t biased. We tried downsampled the “not selected” class but that was not helpful.
* In this dataset we have used Synthetic Minority Over Sampling Technique (SMOTE). It is a standard approach when dealing with unbalanced data and helps us get a more balanced data. Furthermore, it helps avoid overfitting.
* One mistake that we were making earlier was to undersample and oversample the data before cross-validation (CV). This wasn't intuitive and took some pondering before we figured out where we were going wrong. It seem obvious in retrospect if we think about how CV works. For instance, if splits = 3, 1/3 is validation set and 2/3 is used to train. Ideally, the 1/3 should not be touched before we start the training process. This is not preserved if we perform upsampling/downsampling before CV. The phenomena is called "data-leakage" and is explained over at [Machine Learning Mastery](<https://machinelearningmastery.com/data-leakage-machine-learning/>)
* In addition to the upsampling, we normalized our data using the Standard Scalar()

**PART II: Classification**

**Model One:**

**1. Supervised learning method used in this model is**

Support Vector Machines

**2. Why you choose this supervised learning method?**

1. Going with SVMs made sense both intuitively and theoretically. SVMs provide the ability to map our feature space in a higher dimension, separating the data with a margin as wide as possible.
2. Given a binary classification problem, this was something we thought would help us separate the data points.

**3. Describe the method you used to evaluate this method.**

1. We have used multiple methods to give us a deeper understanding in order to tune the parameters and clean the data. These metrics include
   1. precision,
   2. recall,
   3. f1 score and
   4. area under ROC curve.

**4. Describe process of experimenting different parameter settings or associated techniques.**

**○ Parameter name:** Kernel

**- Parameter values:** Radial Basis Function(rbf)

**- Performance of different values:**

* The accuracy of test data with Linear Kernel was 49 percent which is less than the accuracy of Radial Basis Function

**- Analysis:**

**●** By using the different kernel we can increase the accuracy on our test data ie: like by using radial basis function.

**○ Parameter name:**Gamma

**- Parameter values:** auto

**- Performance of different values:**

**●** Low Gamma can create problem due to which our model was underfitting and our regularization parameter value is low to avoid this we use gamma value which is medium

**- Analysis:**

**●** High gamma can lead to low regularization value which can create problem and leads to overfitting.

**5. Accuracy and Confusion matrix with most suitable parameters**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Predicted** |  |
|  |  | **YES** | **NO** |
| **Correct** | **YES** | **9820** | **8170** |
|  | **NO** | **728** | **1500** |

**Model II:**

**1. Supervised learning method used in this model is**

Random Forest Classifier

**2. Why you choose this supervised learning method?**

1. Random forest was one of the models we were looking up to. Practically, decision trees are excellent at handling imbalanced datasets and performing well when the faced with balance they haven’t seen before while training due to an attribute called class weights.
2. An additional advantage is the quick running time while testing (practical deployment) and relatively higher accuracy which can be achieved by considering multiple decision trees, i.e. a random forest. It was ideal for our binary classification problem.

**3. Describe the method you used to evaluate this method.**

We have used multiple methods to give us a deeper understanding in order to tune the parameters and clean the data. These metrics include

* precision,
* recall,
* f1 score and
* area under ROC curve.

**4. Describe process of experimenting different parameter settings or associated techniques.**

There were a number of parameters to play with when it comes to Random Forests Classifiers. We tested a number of them, tweaking around till we got the current performance.

**○ Parameter name: n\_estimators**

**- Parameter values: 200**

**- Performance of different values:**

**●** The default value of the estimator is 50 in sklearn. At that level our accuracy on test was 33.38957857% and f1 score was 0.11.

**- Analysis:**

* One of the tell signs of overfitting is a high training accuracy and a low test accuracy. Our guess was that the model had high bias and/or highly correlated trees. The fix for both is increasing the number of estimators. After tweaking about a bit, we settled on as the ideal number which gave the best results.

**○ Parameter name: criteria**

**- Parameter values: “entropy”**

**- Performance of different values:**

* The default value of criteria in sklearn is gini. The first time we ran our model, we were looking for a base case and didn’t change it, respecting both gini and entropy have their advantages.

**- Analysis:**

* Since we were upsampling our classes before the training, the training data would not be unbalanced. This seemed like a use for entropy.
* As we know, the entropy is 0 if all samples at a node belong to the same class and maximum when we have a uniform distribution.
* Thus a group with 50-50 distribution seemed like better for entropy rather than gini.

**5. Accuracy and Confusion matrix with most suitable parameters**

**Accuracy: 0.7913570722707195**

**F1 score: 0.25**

**AUC: 0.5794249039891491**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Predicted** |  |
|  |  | **YES** | **NO** |
| **Correct** | **YES** | **15427** | **2566** |
|  | **NO** | **1578** | **681** |

**Model III:**

**1. Supervised learning method used in this model is**

Multi Layer Perceptron

**2. Why you choose this supervised learning method?**

1. MLPs are suitable for classification prediction problems where inputs are assigned a class or label.
2. They are very flexible and can be used generally to learn a mapping from inputs to outputs.
3. They can also used as performance evaluation matrix to see how other model are performing.

**3. Describe the method you used to evaluate this method.**

1. Hidden layers
2. Number of Iterations
3. Learning Rate
4. Relu Activation Function

**4. Describe process of experimenting different parameter settings or associated techniques.**

We first decide the number of hidden layers to use as they have a major impact on the performance of the model.Addition to that we have used which activation function to use to train our model.

MLPClassifier(max\_iter = 100, activation = 'relu', hidden\_layer\_sizes = (13, 13, 13), learning\_rate = 'constant', alpha=0.001, solver = 'adam', early\_stopping = True)

**○ Parameter name: hidden\_layer\_sizes**

**- Parameter values: (13, 13, 13)**

**- Performance of different values:**

**●** (20, 20, 20) , (10, 10), (20, 15, 10) {in that order}

**- Analysis:**

* Hidden layers are probably one of the most difficult parameters to tune down. (13, 13, 13) is the default value of hidden layers in sklearn. We changed that in the beginning to a larger number of node out of curiosity.
* We found that the gradients were too small for majority of the nodes in layer 1. This was surprising as we didn’t expect to see a vanishing gradient problem in such a shallow network. We believe this was caused due to the small number of input nodes and a lot of 0 values in the dataset.
* We decreased the size of both layer and came down a layer in size. That didn’t improve our metrics. We then decided to

**Parameter name: learning\_rate**

**- Parameter value: 0.001**

**- Performance of different values:**

**○ Parameter name: activation, early\_stopping, solver**

**- Parameter value: ‘relu’, ‘Yes/True’, ‘Adam’**

**- Performance of different values/analysis:**

* We didn’t try other values/solvers and functions for these parameters. They are widely accepted, work surprisingly well in most cases and need not be tweaked around with till all other options have been exhausted or the task demands.

**5. Accuracy and Confusion matrix with most suitable parameters**

**Accuracy: 0.5265499235872417**

**F1-score: 0.24254411435682935**

**AUC: 0.6098985373461968**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Predicted** |  |
|  |  | **YES** | **NO** |
| **Correct** | **YES** | **9833** | **8160** |
|  | **NO** | **738** | **1521** |

**PART III: Best Hypothesis (No more than two pages for this part)**

**1. Which model do you choose as final method?**

**Model number: 2**

**Supervised learning method used in this model: Random Forest**

**2. Reasons for choosing this model.**

1. The model gave us the highest f1 score and accuracy on the test data. This meant that the model had generalized well and wasn’t underfitting/overfitting.

**3. What are the reasons do you think that make it has the best performance?**

1. Random forest was one of the models we were looking up to. Practically, decision trees are excellent at handling imbalanced datasets and performing well when the faced with balance they haven’t seen before while training due to an attribute called class weights.
2. An additional advantage is the quick running time while testing (practical deployment) and relatively higher accuracy which can be achieved by considering multiple decision trees, i.e. a random forest. It was ideal for our binary classification problem.
3. Furthermore, its hyperparameters can be intuitively tuned.