DATA-448 In-Class Assignment March 27, 2023

Consider the `turnover.csv` data file (posted under the In-Class 18 assignment link). This file contains basic employment information of employees from some company. The goal is to build a binary classification to predict employee turnover. **In Python**, answer the following:

1. (3 points) Using the pandas library, read the csv data files and create three a data-frame called `turnover`.
2. (5 points) Change `sales`, and `salary` from labels to dummy variables.
3. (5 points) Engineer the interactions/features from in-class 9 assignment.
4. (5 points) Using `satisfaction_level`, `last_evaluation`, `number_project`, `average_montly_hours`, `time_spend_company`, `Work_accident`, `promotion_last_5years`, `sales` (dummy variables), and `salary` (dummy variables) and interactions/features (from part 3) as the input variables and `left` as the target variable, split the data into two data-frames (taking into account the proportion of 0s and 1s) `train` (80%) and `test` (20%).
5. (10 points) Based on the different models built on this dataset, it seems that `interaction_3`, `interaction_1`, `satisfaction_level`, `time_spend_company`, and `number_project` are the top 5 important variables. Using `train` data-frame and the top 5 features, perform a hyper-tuning job on the random forest model. Using the `GridSearchCV` function and the following dictionary:

   ```
   RF_param_grid = {'n_estimators': [100, 300, 500],
                    'min_samples_split': [10, 15],
                    'min_samples_leaf': [5, 7],
                    'max_depth' : [3, 5, 7]}
   ```

   perform the hyper-parameter job with 3 folds. Note that based on historical data, the company estimated the following:

		Actual Class	
		0	1
Predicted Class	0	\$0	-\$1,000
	1	-\$1,500	\$500

   Using the information from the above table, the cost function is given by:

   $$\texttt{cost} = -1000 \times Y - 1500 \times Z + 500 \times W$$

   where Y, Z and W represent the number of times the model predicted 0 and it was actually 1, number of times the model predicted 1 and it was actually 0, and number of times the model predicted 1 and it was actually 1, respectively. Identify the hyper-parameter combination that produces the highest cost. Then, use that model to predict the likelihood of `left` on the `test` data-frame. Find the optimal cutoff value by comparing the likelihoods of `left` in `test` and the actual `left` values in the `test`. Use this cutoff to change the likelihoods of `left` in the `test` data-frame to label. Compute the cost of this prediction on the `test` data-frame.
6. (10 points) Based on the different models built on this dataset, it seems that `interaction_3`, `interaction_1`, `satisfaction_level`, `time_spend_company`, and `number_project` are the top 5 important variables. Using `train` data-frame and the top 5 features, perform a hyper-tuning job on the XGBoost model. Using the `GridSearchCV` function and the following dictionary:

```
XGBoost_param_grid = {’n_estimators’: [500],
                      ’max_depth’: [3, 5, 7],
                      ’min_child_weight’: [5, 7],
                      ’learning_rate’: [0.01],
                      ’gamma’: [0.3, 0.1],
                      ’subsample’: [0.8, 1],
                      ’colsample_bytree’: [1]}
```

perform the hyper-parameter job with 3 folds. After that, build a XGBoost model with the best hyper-parameter combination that produces the highest cost (defined in part 5). Then, use that model to predict the likelihood of `left` on the `test` data-frame. Find the optimal cutoff value by comparing the likelihoods of `left` in `test` and the actual `left` values in the `test`. Use this cutoff to change the likelihoods of `left` in the `test` data-frame to label. Compute the cost of this prediction on the `test` data-frame.

7. (3 points) Based on your results from parts 5, and 6, what model would you use to predict `left`? Be specific.

The End.