COMP4432 Machine Learning - Assignment 3

Learning Objective: This assignment will offer you the opportunity to work through data evaluation and cleaning, and construct a few classifiers to predict whether a passenger survived their trip on the Titanic. The feature *survived* is the Target variable for this analysis. The dataset employed is from the Seaborn library, and includes some interesting nuances which you will explore.

Part 1: Data Exploration and Preparation

a. Load the Titanic DataFrame from *Seaborn* using the *load_dataset('titanic')* method.
b. For all features in the DataFrame, identify the number of instances with null values and the percentage of the feature that is null. Hint: one feature is mostly nulls (> 75%).
c. Remove the feature that is mostly null values.
d. This dataset includes four redundant features, that is separate features that represent equivalent information (*alive* & *survived*, *class* & *pclass*, *embarked* & *embark_town*, and *adult_male* & *who*). Conduct and present quantitative analysis to verify these four pairings.
   Hint: There is no single correct technique. Trying Pandas *drop_duplicates* or *value_counts* is pretty straightforward. Pandas *groupby* is also useful.
e. Drop the *alive, class, embarked_town,* and *who* input features from the dataset.
f. The feature *adult_male* was engineered from other features remaining in the input dataset. First, conduct and present analysis to identify the logic to construct *adult_male* from other features in the data set. Second, in a Markdown cell, present the logic to construct *adult_male*. The logic should resemble: "*adult_male* is engineered from features X and Y, such that if X is greater than, or equal to BLAH, and Y is equal to BLEH, then *adult_male* is True, else *adult_male* is False.
   Hint: The feature name should offer a clue to the features employed. Also, consider the *groupby* functionality.
g. Drop the *adult_male* input feature from the DataFrame.
h. Explore the data and answer the following questions in markdown cells:
   i. Did more women or men ***die*** on the Titanic?
      Show this as 1) a count by *sex*, and 2) a percentage by *sex* (that is, of the total number of male passengers, what percentage of males died? And for the total number of female passengers, what percentage of females did not survive?)
   ii. What percentages of passengers ***survived*** by passenger class?
   iii. Describe the distribution of fare values
   iv. Describe the distribution of non-null age values
   v. What is the median age for each passenger class?
   vi. Plot the distributions of age by passenger class in box plots.
i. Drop the rows that are missing values in *embark_town,* and show the number of rows remaining in the data set.
   Hint: Review the *dropna* function documentation to identify how to limit the removal to a set of features. Also, 889 rows should remain.
j. Partition the data set into Training and Testing sets. Set aside 20% of the data for testing, and stratify on the target variable to maintain the imbalance in Training and Testing sets.
k. Show that the target rate was maintained between the entire data set, Training data, and Testing data.
   Hint: Target rate is the percentage of one labels in a binary target data set.
l. Null values are still present in the *age* input feature. From the Training Data, construct a stratified imputer that will replace missing *age* values with the median age by the passenger class. Employ this imputer to replace missing *age* values in the Training and Testing Data sets.
   Hint: Pandas *groupby* functionality is really helpful.

m. Using either *.info()* or *.dtypes*, identify and document which features in the Training Data are of Pandas data type *category*.
n. Review the data within the Training Data set, and in a Markdown cell document which input features could be categorical so that they can be encoded.
   Please keep in mind that the feature *survived* is the Target variable, so it won't need encoding.
   Hint: The feature *pclass* has a data type *int*, but only takes on three unique values. Is this feature really an *int*, or is it *categorical*? Examine the unique values each feature can take, and consider the feature name. I found four input features that were categorical.
o. Using the *OneHotEcoder* functionality in *sklearn.preprocessing*, encode the identified categorical features in the Training and Testing data set.
   Hint: Set *sparse_output = False* to get dense output.
   Hint2: Review the shared notebook: *one_hot_encoding.ipynb*

Part 2: Initial Model Training

a. Create instances of a logistic regression model, a support vector classifier (set *probability = True* upon instantiation), and a decision tree classifier into appropriately named variables. Do not fit these classifiers yet.
b. Using the *cross_val_predict* functionality from *sklearn.model_selection* and the training data sets from Part 1n, calculate the probability predictions from each of the three classifiers. Store the results into appropriately named variables.
   Hint: Use *method='predict_proba'* for all three algorithms.
c. Print a classification report, confusion matrix, and ROC-AUC score for each classifier.
   Hint: Classification report and confusion matrix want class labels for the *y_pred* parameter. The output of *cross_val_predict* is in terms of class 0 and class 1 probabilities. Consider how to identify class label predictions from these probabilities.

Part 3. Model Tuning

a. The training and testing data was not standardized in previous steps. Employing StandardScaler, fit and transform the training and testing data sets following the protocol discussed and demonstrated in the live sessions.
b. Using *cross_val_predict* with *method='predict_proba',* the instantiated support vector classifier from Part 2a, and the scaled Training data, recalculate the probability predictions for the Support Vector Classifier, and print the classification report, confusion matrix, and ROC-AUC score.
   Hint: Classification report and confusion matrix want class labels as the *y_pred,* and the output of *cross_val_predict* is in terms of class 0 and 1 probabilities.
c. In a Markdown cell, compare and document the differences between evaluation metrics calculated from the Support Vector Classifier with unscaled and scaled Training data.
d. Using *GridSearchCV*, the scaled Training data, an ROC-AUC scoring metric, and the Support Vector Classifier, conduct a grid search using the following parameter space:
   {'kernel': ['rbf'],
   'gamma':[0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 2, 5],
   'C':[0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10]}
e. From the fitted *GridSearchCV* object, print the best parameters and the corresponding best score.
f. With the best estimator from the fitted *GridSearchCV* object, make predictions with the scaled Testing data set, and present the classification report, confusion matrix, and ROC-AUC score.