Code	RANDOM_FOREST.PY
Author	Nathaniel Heatwole, PhD (heatwolen@gmail.com) (GitHub) (LinkedIn)
Summary	Uses a random forest to predict survival for passengers in the Titanic disaster.
Methods/ process	Random forest: - Supervised learning method that fits many decision trees ("forest") and aggregates the results. - Combines benefits of decision tree learning, while mitigating their tendency to overfit to their training data. - Each decision tree fitted on: 1) random subset of features; and 2) random selection of training data observations (with replacement). - Randomly withholding some information (that would otherwise be available to fit the model) reduces correlations between the trees. - Trees can be split using various measures, including entropy[1] and Gini impurity[2] (minimum sought in either case). - *Root node* (top of tree): quantity/threshold yielding the best split. - Predictions of the many individual trees are homogenized using either plurality vote (classification) or average (regression). - Out-of-bag testing can also be used (if entire dataset not used to generate tree). Steps: 1. Import and clean training data. 2. Conduct feature engineering (prepare data for use in a random forest model and maximize useful information to be extracted from it). 3. Assess feature importance (Pearson correlation matrix, chi-squared, and coefficient of variation). 4. Fit random forest to training data to predict survival.
Training data	Titanic dataset – containing data for 891 Titanic passengers (from Kaggle).
Output	Summary - Feature importance Plot - Actual versus predicted survival (with group-level summary stats)
Result	The predictions align with expectations: high predicted survival probabilities for survivors (mean: 0.87), and the opposite for non-survivors (mean: 0.09). Additionally, the predicted probability IQRs for the two groups are non-overlapping. Feature importance indicates fare, male, and age are the most consequential variables.

[1] *Entropy* is a measure of disorder. It is defined as the average (expected value) of information, and is equal to: sum[-p * ln(p)].

[2] *Gini impurity* measures how often a randomly chosen element would be incorrectly labeled (if group labels were assigned randomly using the distribution of labels in the training set). It is equal to: sum[p * (1 - p)].