Coursework Assignment 1 – Individual (Summary Report)

This project focuses on the use of machine learning methods in order to successfully train models using customer information for making predictions on whether a customer will make a fixed term savings deposit (FTSD) or not (i.e. a binary classification problem (BCP)). The original dataset (bank marketing data from the UCI Machine Learning Repository) contains 41188 rows with 19 features (economic/ customer information that models can base predictions off) as well as an outcome column (label detailing if the customer made a FTSD).

Data exploration involved observing the shape/ structure of the dataset, plotting, and examining relationships between columns. It was determined that the dataset is very imbalanced with regards to the outcome column (~7x more "no" than "yes"). This is important to note, especially when it comes to preparing the dataset for model training and choosing evaluation metrics. Bar plots were created for categorical columns highlighting the split between the two outcomes ("yes"/ "no") for each item in a column. This was done to grasp an idea of which attributes are the largest predictors of customers making FTSDs. Careful considerations were made so as not to leak information from our outcome class when using ordinal encoding (a transformer that converts categorical information into numerical, preserving inherent ordering), such as not ordering jobs based of off which have the largest "yes" ratios in plots. Histograms for numerical data were utilised to gain insights into the distribution of the numerical columns. Additionally, correlations were calculated between the numerical columns, which helped develop understanding on how they interact with each other (useful for determining if any may be redundant or should be dropped). Missing values were also identified, all in categorical columns (defined as "unknown").

To prepare the dataset for our algorithms, the duration of call column was dropped as suggested by an included information file for a more realistic model. Additionally, in the number of days passed since last contact column (C*), values of 999 were altered to -1 as 999 was a placeholder indicating no contact – the switch helps to clarify this and avoid model distortion during from the large values. Three features were also engineered (transferring raw data into more meaningful outputs): previous contact (derived from C* – more definitive for predictions), job stability (maps jobs to a score – job stability is a positive indicator of the ability to make a FTSD) and financial stability (product of age with job stability score – an estimate of a customer's overall financial state).

Multiple routes were investigated for optimal data preparation: combinations of imputing values (filling in missing values, commonly done with the mode for categorical), dropping columns and/ or rows containing missing data, and leaving the "unknown" occurrences in. The options considered are discussed more in depth in the notebook but ultimately it was decided that "unknown" data would be preserved in the dataset and encoded. This is because in the real world, missing data is fairly common and it makes sense to train a model which is capable of dealing with it, especially for data like this where for the user submitting the information there is likely a "prefer not to answer" option. Data was split into training (85% of data) and test (15% of data) sets using stratified sampling for preservation of the distribution of our outcome column (better for imbalanced classes).

Encoding methods were also experimented with (1. one-hot with ordinal vs. 2. just one-hot) alongside the previous methods mentioned. When using method 1 categorical columns were split into those with natural order (for ordinal, e.g. Mon, Tue, Wed) and those without (for one-hot, e.g. telephone, cellular). For optimising our model performance, it was determined that the use of just one-hot encoding (converts categorical data into a binary vector) was best (especially when not discarding missing data, which is likely due to the inherent difficulty of ordinally encoding unknown values). Numerical values were scaled. This combination of encoding and handling missing data was determined via evaluating models and examining relationships from plots.

Macro average (MA) F_1 score was the primary metric used for evaluation and fine-tuning of hyper-parameters (settings which tweak how the model learns/ functions). This metric was chosen due to it being well suited for use with BCPs and imbalanced datasets. To optimise predictions, multiple models were trained and evaluated, namely, stochastic gradient descent (SVM), logistic regression (LR) and random forest (RF). Initial fits using cross-validation (CV) with stratified KFold were performed (ensures that folds maintain the same distribution of outcome labels, good for imbalanced classes). Results from our initial CVs were fairly similar, so each model was fine-tuned using a grid search using more CV to attempt to identify a definitive best model for the purposes of the problem. Each fine-tuned model was evaluated on the unseen test set (exact models and scores in Table 1) and had a ROC and Precision-Recall curve created (Fig. 1) which are useful in the evaluation process. RF was our best performing model by our main metric, achieving a MA F_1 of 0.73 on the test set, making it well suited for answering this BCP given the imbalance of the dataset. If computational or time restrains are present then it may be better to select one of the slightly weaker, but faster SVM or LR which achieve a MA F_1 of 0.69.

Appendix A

Fine-tuned Model	*P0*	*P1*	*R0*	*R1*	*$F_1$0*	*$F_1$1*	*MA F_1*
SVM: {'alpha': 0.001, 'max_iter': 500, 'penalty': 'l1'}	0.95	0.37	0.87	0.67	0.91	0.47	0.69
LR: {'C': 500, 'max_iter': 100, 'solver': 'lbfgs'}	0.95	0.37	0.86	0.65	0.90	0.47	0.69
RF: {'bootstrap': True, 'max_depth': 20, 'min_samples_leaf': 2, 'min_samples_split': 10, 'n_estimators': 300}	0.94	0.5	0.93	0.57	0.94	0.53	0.73

Table 1. Model performances on unseen test set (P = Precision, R = Recall, F_1 = F_1 Score, 0 = "no", 1 = "yes")

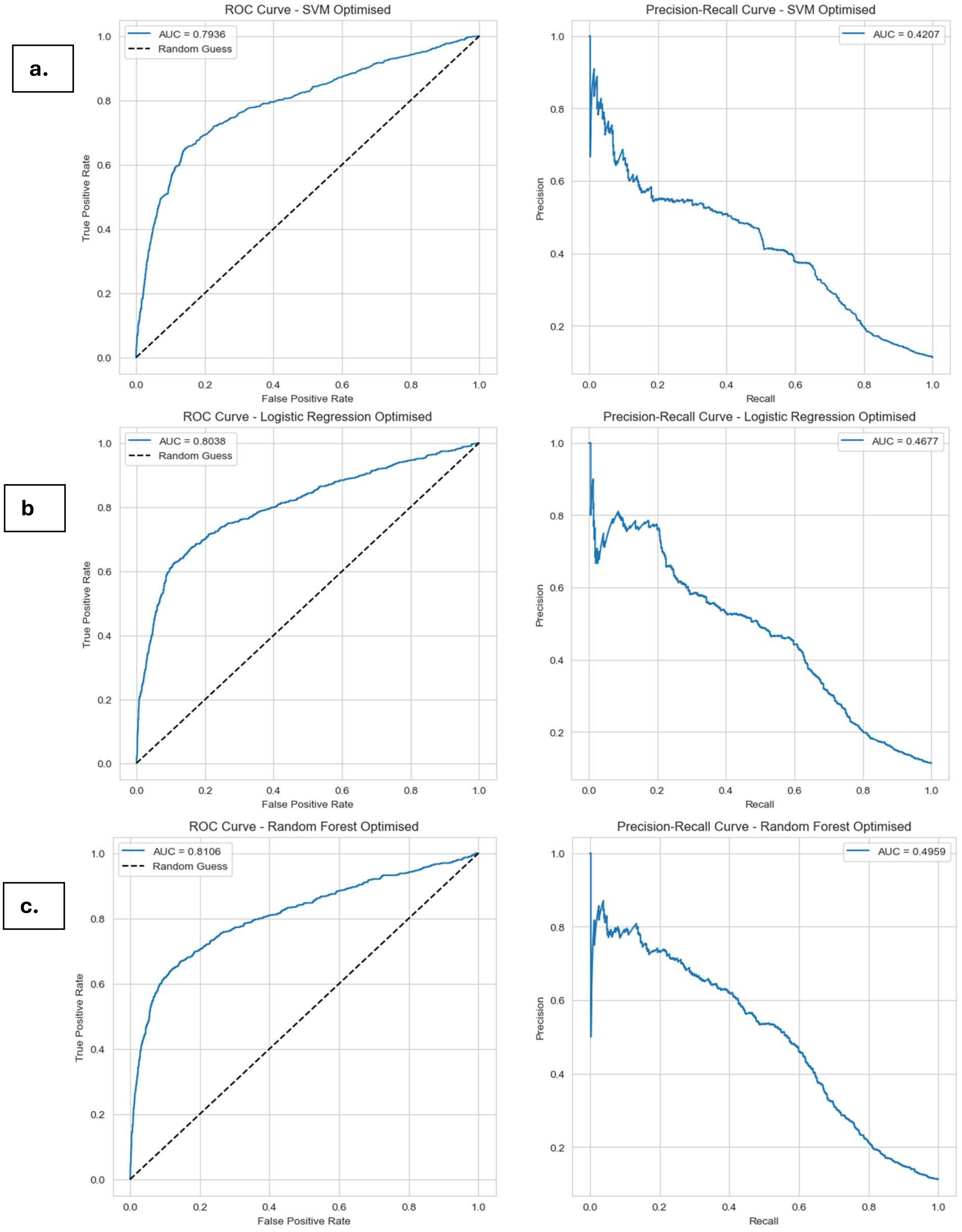

Figure 1. ROC curves (left) alongside Precision-Recall curves (right) for models **a.** SVM, **b.** LR, **c.** RF