Final Project Written Report

CS-UY 4563

11:00AM Section

05/02/2022

Prof. Linda M. Sellie

Alan Wu, Claire Huang

Introduction

The data set we chose to use is hotel booking demand from Kaggle. It was originally from the article Hotel Booking Demand Datasets, written by Nuno Antonio, Ana Almeida, and Luis Nunes for Data in Brief, Volume 22, February 2019. This dataset contains 13930 booking information from a city hotel and a resort hotel. Each data contains information including a year of arrival date, the month of arrival date, code of room type reserved, ID of the travel agency that made the booking, when the booking was made, length of stay, etc. The target we used for the training is whether the hotel booking was canceled or not. There are 32 features including the target in the original dataset.

Through the hotel booking demand database, we were able to make different learning models based on information in the data to help the hotel make predictions on whether the customer would have the possibility to cancel the reservation to make sure that they can best avoid the possibility of overbooking but earn the highest benefits at the same time. Our goal was to use logistic regression, support vector machines (SVM), and neural networks to predict the likelihood of cancelation for the hotel among the 13930 samples, which involves calculating the training accuracy, and the testing accuracy for every model. For each data, we adjust the value of the hyperparameter for different learning models ranging from 0.001 to 100.

Unsupervised Analysis

We can discover some interesting patterns from the data by looking at the distribution of number of bookings. We mainly examined the distribution with respect to months and countries.

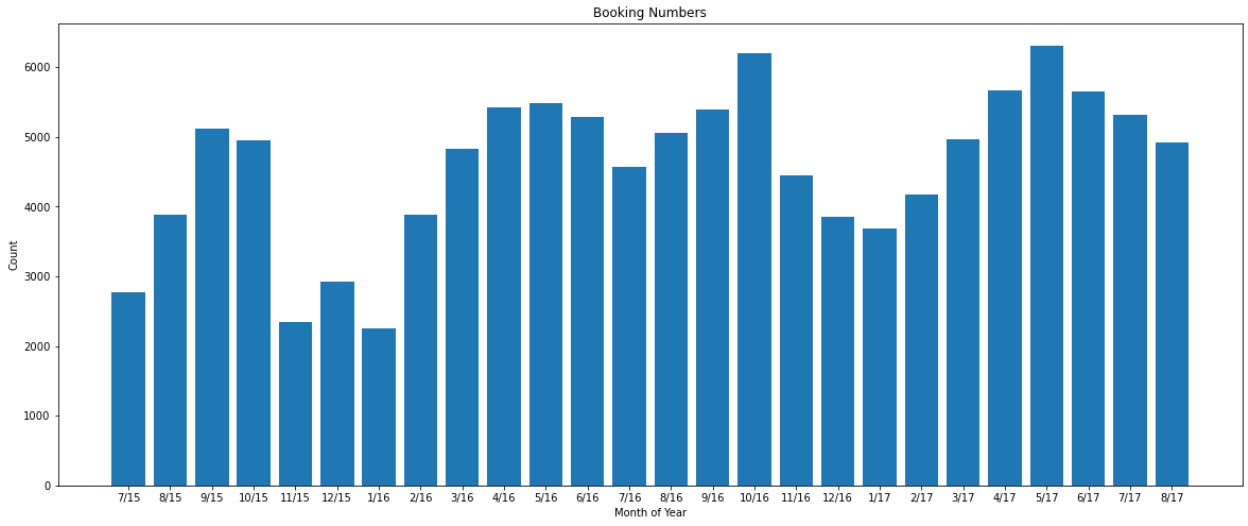


Table 1. Booking Numbers Distribution

It is interesting to see that the peaks of the booking number occurs around May and October, while the valley occurs around January and July. The peaks may correspond to local holidays, and the valley around January may be due to Christmas when people tends to stay with their families at home.

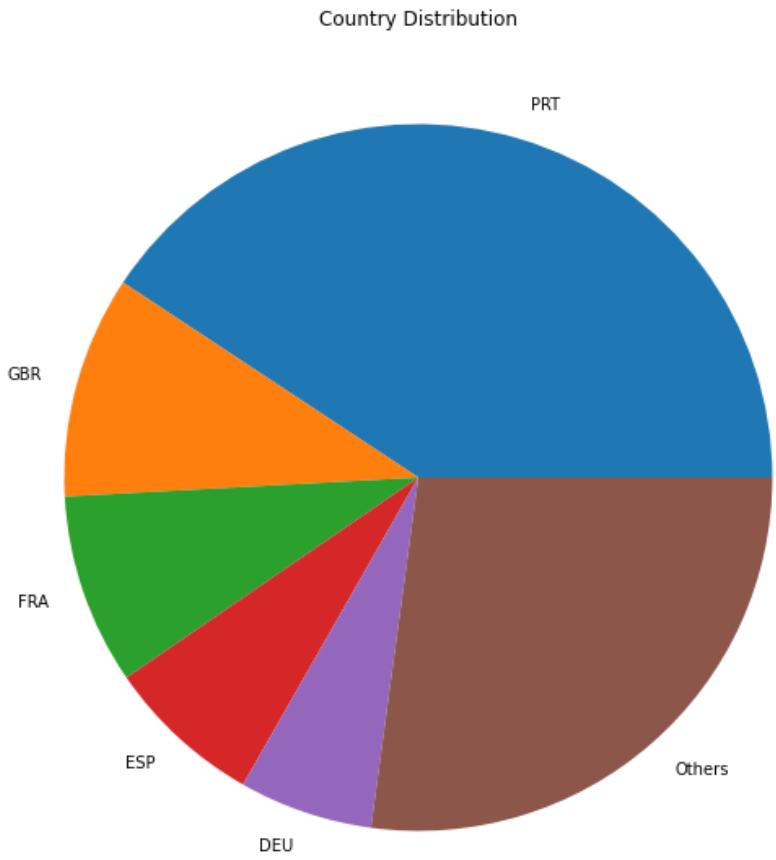


Table 2. Country Distribution of Bookings

The country distribution of the number of bookings shows that the hotels may locate in Portugal, and they receive more bookings from countries nearby than those that are farther.

Preprocessing

Before starting training, we had to clean the data and choose the features we would like to use. The original data has 32 columns, including 1 column “is_canceled” that is used as a prediction target. We examined each feature and decided how to transform it and whether to use

it. For numerical features we keep it and wait for scaling before putting it into the model. For text features with not too many categories, we used one-hot encoding to transform them to numbers. We did not use the date information (including year, months of year, day of month, week of year, etc) because these numbers do not have a relative relationship in size, but transforming them using one-hot encoding would add too many features and increase the probability of overfitting. For “reserved_room_type” and “assigned_room_type”, they each have around 10 categories, where one-hot encoding would also add too many features. After observation, we discovered that around 87% of the guests get the same room type they reserved, so we decided to combine the two features to a bool type indicating whether the reserved and assigned room types match. All data are scaled using the StandardScaler from sklearn before put into training.

Logistic Regression

After preprocessing the data, we reduced the data with missing features and we first tried the LogisticRegression model discussed in the class from the “sklearn.linear_model” library. We first started with a base model without any penalty. The training accuracy for the regularization without any penalty is about 0.815823 and the test accuracy for the regularization without any penalty is about 0.811539. We then tested both models with L1 lasso regularization and L2 ridge regularization. For each of the regularizations, we tested the model with different hyperparameter values of [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]. Then for both the L1 and L2 regularization, we performed additional polynomial feature transformation. The results of these regularizations and regularizations with transformations are shown below(Figure 1 & Figure 2). Each graph shows the training and test accuracies of different logistic regression models.

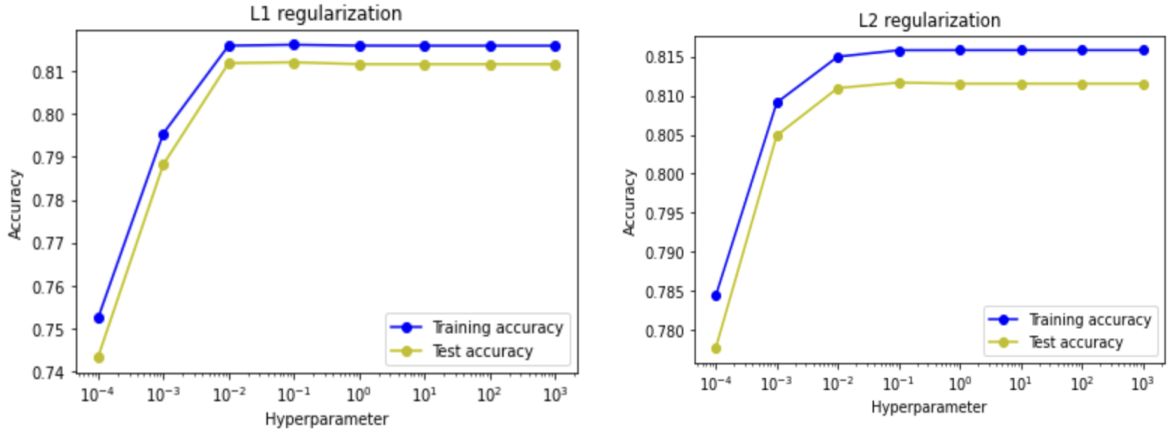


Figure 1: The training and test accuracies of Logistic Regression for L1 and L2 (C range from 0.0001to 1000)

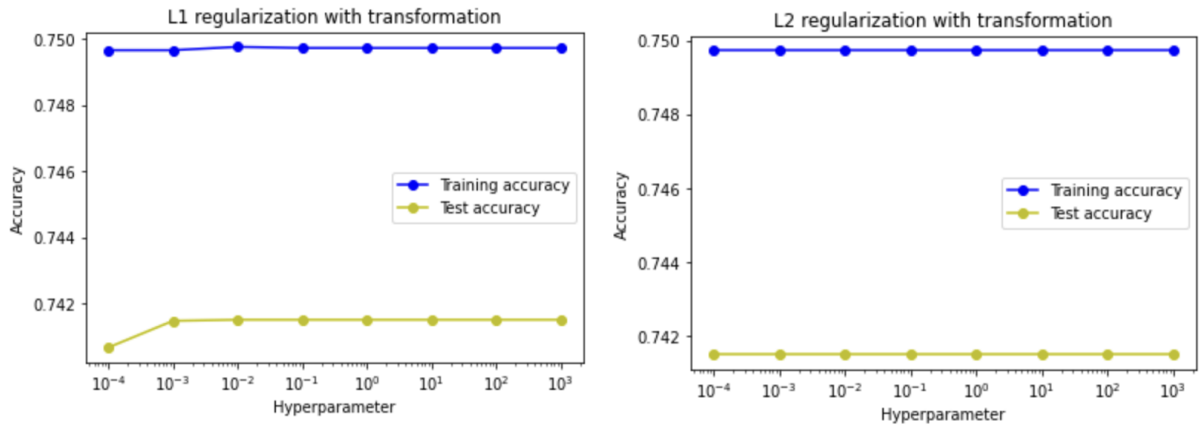


Figure 2: The training and test accuracies of Logistic Regression for L1 and L2 with Polynomial Feature Transformation (C range from 0.0001to 1000)

Support Vector Machine (SVM)

After performing the data with the Logistic Regression Model, we then move to the SVC model discussed in class by creating an SVC object from the "sklearn.svm" library. For each performance, we try different hyperparameters and kernels. We tried three different kernels: the linear, polynomial, and radial-basis function kernel. For each kernel, we tested the model with different hyperparameter values of [0.0001, 0.001, 0.01, 0.1, 1, 10, 100]. Since we noticed that the difference between the training accuracy and the test accuracy is getting slightly greater as the hyperparameter increases, we realized that it is slightly overfitting so we decided to reduce the max of the hyperparameter from 1000 to 100. Moreover, since we have about 119390 samples, the SVM training seems to take an overwhelming amount of time so we decided to reduce the sample to about 2000 to perform the data. The samples were randomly chosen from the dataset through the train_test_split method.

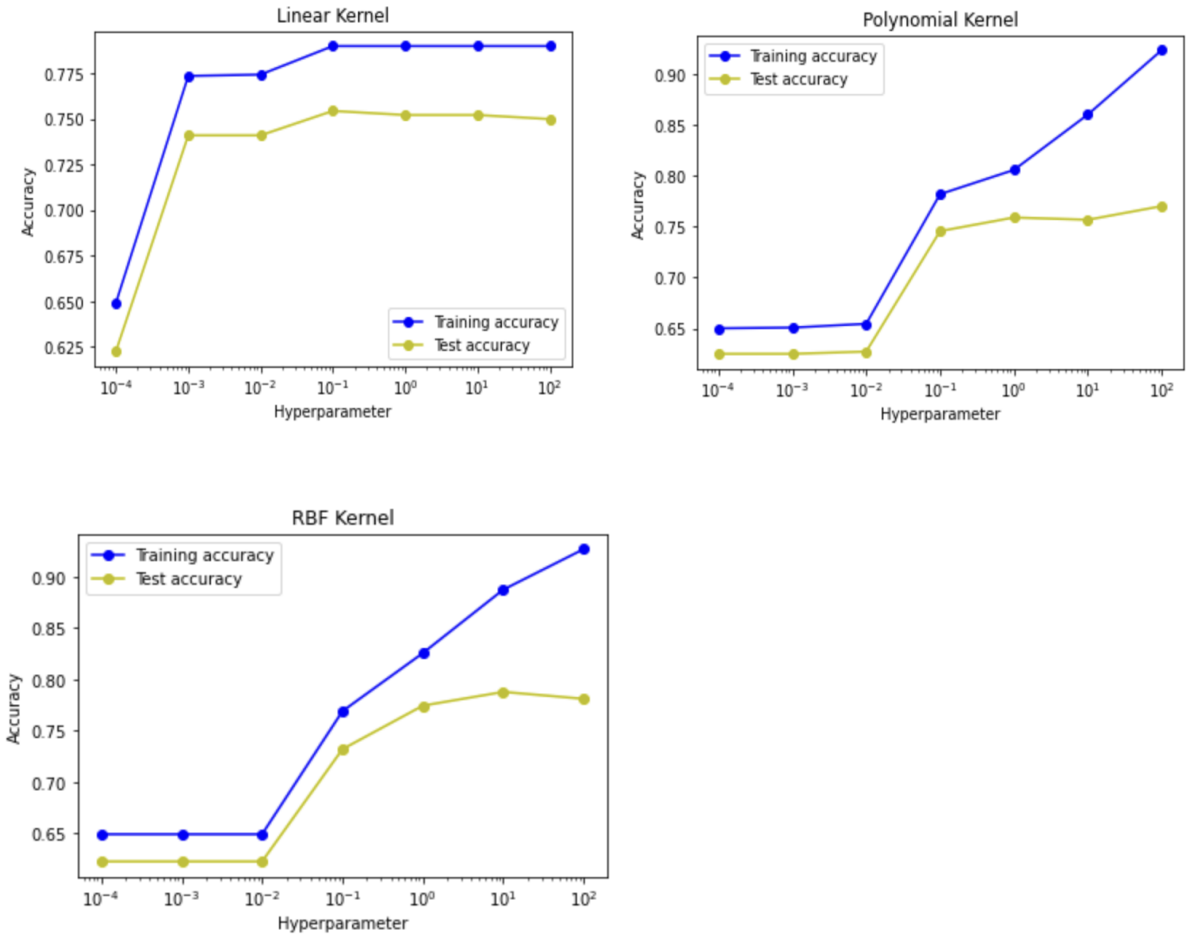
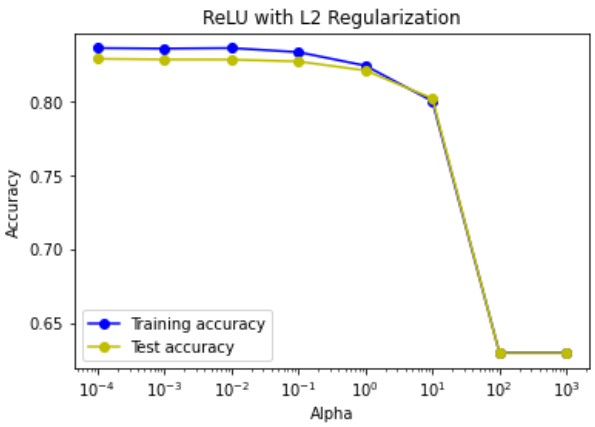
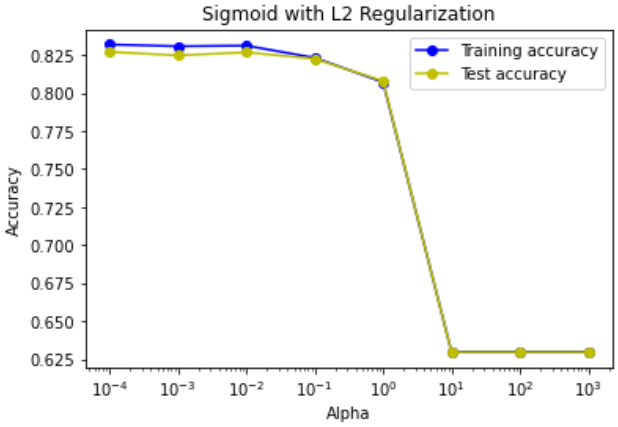


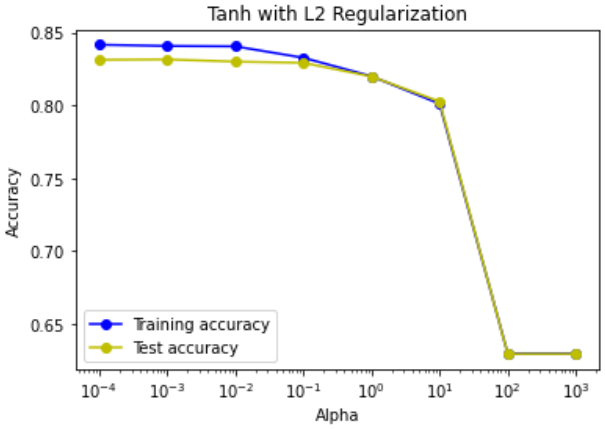
Figure 3: The training and test result of SVM for Linear Kernel, Polynomial Kernel, and RBF kernel

Neural Networks

For hidden layer settings, based on test accuracy of models trained under the same conditions (ReLU, hyperparameter, training and test sets), we discovered that neural networks with two hidden layers generally have higher accuracy than those with only one; and accuracy is similar for hidden layers from (20,8) to (30,10). So we decided to use a neural network with two hidden layers (20,10), i.e. the overall shape is (42,20,10,1). The output layer has only one neuron of value either 0 or 1 as prediction. We tested the model with L2 regularization with different hyperparameter values of [0.0001, 0.001, 0.01, 0.1, 1, 10, 100], and tried different activation functions including ReLU, sigmoid, and tanh. The size of the training set was chosen to be 40% of all the data to reduce time of training.







Tables of Results

	0.0001	0.001	0.01	0.1	1	10	100	1000
None, None, Train	0.815823	0.815823	0.815823	0.815823	0.815823	0.815823	0.815823	0.815823
None, None, Test	0.811538	0.811538	0.811538	0.811538	0.811538	0.811538	0.811538	0.811538
L1, None, Train	0.752498	0.795418	0.815834	0.816068	0.815845	0.815823	0.815823	0.815823

L1, None, Test	0.743424	0.788219	0.811773	0.811974	0.811538	0.811538	0.811538	0.811538
L2, None, Train	0.784328	0.809111	0.814985	0.815812	0.815823	0.815823	0.815823	0.815823
L2, None, Test	0.777599	0.804905	0.811036	0.811672	0.811538	0.811538	0.811538	0.811538
L1, transformation, Train	0.749650	0.749651	0.749751	0.749717	0.749718	0.749718	0.749718	0.749718
L1, transformation, Test	0.740677	0.741481	0.741515	0.741515	0.741515	0.741515	0.741515	0.741515
L2, transformation, Train	0.749717	0.749717	0.749717	0.749718	0.749718	0.749718	0.749718	0.749718
L2, transformation, Test	0.741515	0.741515	0.741515	0.741515	0.741515	0.741515	0.741515	0.741515

Table 1: Table of results for Logistic Regression; Top rows are C-values; Left columns indicates the detail of the model type (L1 or L2, Transformation/ None, Train/Test accuracy); Training data were highlighted in orange and test data were highlighted in red

	0.0001	0.001	0.01	0.1	1	10	100
Linear, Train	0.649292	0.773641	0.774385	0.790022	0.790022	0.790022	0.790022
Linear, Test	0.622767	0.741071	0.741071	0.754464	0.752232	0.752232	0.750000
Poly, Train	0.650037	0.650781	0.654504	0.781831	0.805658	0.860014	0.923306

Poly, Test	0.625000	0.625000	0.627232	0.745535	0.758928	0.756696	0.770089
RBF, Train	0.649292	0.649292	0.649292	0.769173	0.825763	0.887565	0.927029
RBF, Test	0.622767	0.622767	0.622767	0.732142	0.774553	0.787946	0.781250

Table 2: Table of results for SVM; Top rows are C-values; Left columns indicates the detail of the model type (Types of kernel, Train/Test accuracy); Training data were highlighted in orange, and test data were highlighted in red

	0.0001	0.001	0.01	0.1	1	10	100	1000
ReLU, Train	0.836621	0.836285	0.836641	0.833877	0.824684	0.800582	0.629769	0.629769
ReLU, Test	0.829336	0.828889	0.828833	0.827521	0.821420	0.802420	0.629495	0.629495
Sigmoid, Train	0.831993	0.830862	0.831260	0.823260	0.807220	0.629769	0.629769	0.629769
Sigmoid, Test	0.827158	0.824826	0.826851	0.822551	0.808004	0.629495	0.629495	0.629495
Tanh, Train	0.841751	0.840913	0.840641	0.832809	0.820119	0.801231	0.629769	0.629769
Tanh, Test	0.831416	0.831681	0.830145	0.829350	0.819926	0.802783	0.629495	0.629495

Table 3: Table of results for Neural Network; Top rows are alpha-values; Left columns indicates the detail of the model type (Types of activation function, Train/Test accuracy); Training data were highlighted in orange, and test data were highlighted in red

Conclusion

For logistic regression, the training accuracies are generally higher than the test accuracies which are aligned with what we expected. The highest test value for the model is 0.811974 which belongs to the L1 lasso regularization with no transformation for a c-value that is equal to 0.1. So far the model results seem reasonably accurate. The difference between the training accuracies and test accuracies for each model is really small, which is nearly around 0.01 so we assume that there is no noticeable problem with either bias or variance. There should not exist a problem of the overfitting from the results we calculated for both L1 and L2 since it is obvious that the discrepancies between the test and training results for both were getting smaller as we increased the parameter. Compared to models with polynomial features transformation, the model without the transformation performs better. The test accuracy for a model with transformation is around 0.742 which is less than the test accuracy for the model without the transformation. Since we are trying to predict the cancellation of the booking, we did not expect the accuracy to be high but some way to improve the performance could be to increase the sample size and add more relevant features.

For SVM, we noticed that since the sample size is around 119390, it takes much longer time to train the SVM model than we expected so we decided to reduce the sample size to about 2000. From the sample size we selected, we found that the highest test accuracy is 0.787946 for the radial–basis function kernel with the hyperparameter equal to 10. For each kernel we tested, the training accuracy is higher than the test accuracies which are aligned with what we expected. However, the graph strongly demonstrates that there is overfitting. As the hyperparameter increases from 0.0001, we noticed that the difference between training accuracy and test accuracy is getting larger, especially for the polynomial kernels and the RBF kernel. For the linear kernel, the difference between training accuracy and test accuracy is about 0.02 first and then increases up to 0.04. For the polynomial kernel, the difference between training accuracy and test accuracy is about 0.03 first and then increases up to 0.15. For the RBF kernel, the difference between training accuracy and test accuracy is about 0.02 first and then increases up to 0.14. Besides the overfitting, variance or bias seems not found in the model with the sample we randomly chose. Comparing the SVM model to the logistic regression, it is obvious that logistic regression seems to perform better than the SVM. However, since the time for training an SVM model with the original data set seems to be unpredictable, the conclusion cannot be easily drawn. To better improve the SVM, one thing to do is to train the data for a longer time having a machine able to run for days and another way to do it could be using a larger training dataset.

For neural networks, the training accuracies are generally higher than the test accuracies, which is as expected. The highest test accuracy is the one with tanh as activation function and coefficient of 0.001 on L2 regularization terms; but in general the test accuracies have very small differences for alpha between 0.0001 and 0.01. Comparing activation functions we found that in general tanh has the best test accuracy, ReLU the second, and sigmoid the third. Considering we

are using a neural network with 2 hidden layers, it is likely that the low value of derivative of sigmoid decreases the training efficiency; while tanh has a nonzero derivative value for negative inputs, so it uses more information than ReLU, so tanh performs best in this case. Overall, the three different activation functions do not vary significantly in terms of test accuracy. They perform well with hyperparameters between 0.0001 and 0.01, and in general the accuracy of neural networks is generally higher than those of logistic regression and SVM, probably because the more complex model better fits data with a relatively large number of features.

Work Cited

Hotel Booking Dataset:

https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand?resource=download

For plotting graph:

https://www.tutorialspoint.com/how-to-label-a-line-in-matplotlib-python

For logistic regression:

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

For SVM:

https://scikit-learn.org/stable/modules/svm.html

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html