Supervised Machine Learning Models' Performance Comparison

Ayesha Siddiqa Yasmin
Department of Computer Science and Engineering
East West University
2019-3-60-120@std.ewubd.edu

Md. Shajibul Islam
Department of Computer Science and Engineering
East West University
2019-1-60-031@std.ewubd.edu

Mehzabin Meem
Department of Computer Science and Engineering
East West University
2019-2-60-019@std.ewubd.edu

Md. Golam Rabiul Alam
Department of Computer Science and Engineering
East West University
golam.rabiul@ewubd.edu

***Abstract*— Machine Learning is a popular and widely used field in a variety of fields. It is often used in the prediction of classification and regression problems. This study compared and found the highest performing Supervised classification model using SVM, KNN, Adaboost, Random Forest, Voting Classifier, and Deep Neural Networks with and without batch processing. The models were solely used to compare on the hotel reservation dataset, which comprises 36275 instances and 19 characteristics. The performance of the models is highly influenced by the various preprocessing approaches and model types utilized. This study discovered that deep neural networks (DNN) perform the best.**

Keywords*— *Machine Learning, Supervised Learning, Classification, Comparison, KNN, SVM, AdaBoost, RF, VC, DNN.

I. INTRODUCTION

Machine learning is one of the key leading areas in the field of artificial intelligence. Machine learning allows one to train a machine to understand patterns in data and analyze it, as well as learn from and adapt to it.

Because of its numerous applications, including as classification and regression, machine learning is employed in a wide range of fields. Machine learning is further separated into supervised and unsupervised learning due to the great diversity and odd forms of data.

This research focuses on supervised classifications, in which the learner must predict the output class and its pattern based on the input and output. A comparison of supervised classification models will be performed in this work. A few well-known supervised learning models will be used in the comparison. KNN, SVM, Adaboost, Random Forest, Voting Classifier, and Deep neural networks are among them.

K-Nearest Neighbour is a simple machine learning technique based on the supervised learning method. The K-NN algorithm assumes that the new case and the existing cases are comparable, and it assigns the new instance to the category that is most similar to the existing categories.

The Support Vector Machine, or SVM, is a popular supervised learning technique. It is used to tackle problems involving classification and regression. The goal of the SVM algorithm is to find the optimal line or decision boundary for dividing a space with n dimensions into classes, allowing us to swiftly categorize fresh data points. A hyperplane is the optimal choice boundary.

AdaBoost, also known as Adaptive Boosting, is a machine learning algorithm that is employed as part of an ensemble. The most commonly used AdaBoost algorithm is decision trees with one level. It builds a model and assigns equal weight to each piece of data. The points that were mistakenly classified are therefore given higher weights. In the following model, all points with higher weights are given more significance. It will keep training models till the error is reduced.

A random forest, as the name implies, is made up of multiple separate decision trees that collaborate as an ensemble. The random forest generates a class prediction for each tree, and the class with the most votes is the accurate forecast.

A voting classifier is a machine learning model that learns from an ensemble of multiple base models and predicts an output using the output class with the highest likelihood.

Deep neural networks are neural networks with a minimum of two layers and varying degrees of complexity. It processes data in complex ways using advanced math models.

A hotel reservation dataset will be used to forecast if a hotel reservation will be canceled or not. People frequently book a hotel reservation only to cancel it later due to unforeseen circumstances. To reduce the loss, it is preferable if it is feasible to foresee whether or not a customer will cancel their booking in advance.

II. METHODOLOGY

This section describes how we used various supervised machine learning models for training and prediction, as well as the dataset and dataset preprocessing approaches used in this study.

A. DATASET

The Kaggle website was used to collect the hotel reservation dataset. It includes 36275 rows and 19 columns and is 3.08 MB in size. The dataset's columns are as follows:

#	Column	Non-Null Count	Dtype
0	Booking_ID	36275 non-null	object
1	no_of_adults	36275 non-null	int64
2	no_of_children	36275 non-null	int64
3	no_of_weekend_nights	36275 non-null	int64
4	no_of_week_nights	36275 non-null	int64
5	type_of_meal_plan	36275 non-null	object
6	required_car_parking_space	36275 non-null	int64
7	room_type_reserved	36275 non-null	object
8	lead_time	36275 non-null	int64
9	arrival_year	36275 non-null	int64
10	arrival_month	36275 non-null	int64
11	arrival_date	36275 non-null	int64
12	market_segment_type	36275 non-null	object
13	repeated_guest	36275 non-null	int64
14	no_of_previous_cancellations	36275 non-null	int64
15	no_of_previous_bookings_not_canceled	36275 non-null	int64
16	avg_price_per_room	36275 non-null	float64
17	no_of_special_requests	36275 non-null	int64
18	booking_status	36275 non-null	object

We can observe from the preceding data that all of the columns in the data set have non-null values in all 36275 rows. Except for five columns that have object-type values that are basically string values and one column that has float-type values, all of the columns have integer-type values. The 'booking_status' column is the targetclasslabel column that is utilized to forecast and find the accuracies of the models. Other columns are feature columns. The 'booking_status' column has two distinct values: 'Canceled' and 'Not_Canceled,' with 11885 'Canceled' values and 24390 'Not_Canceled' values in the dataset.

B. DATA PREPROCESSING

Data preparation is a critical step that must be completed before using the data to train the machine learning models. It enhances model performance and aids in more accurate prediction of class labels. To begin preprocessing, all extraneous columns ('Booking_ID', 'arrival_year', 'arrival_date') for model training are removed. A heatmap of features correlation is then generated in order to eliminate the strongly correlated features columns in order to lower the features' dimension and model's training complexity.

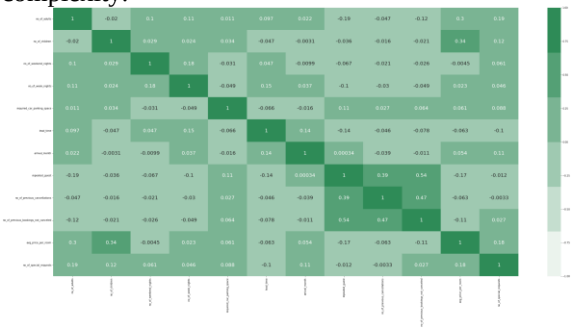


Figure 1: Features' correlation heatmap

We can observe from the heatmap that no feature is highly connected. As a result, no column gets eliminated because it is highly associated with other columns. Then, 10477 redundant rows are removed to improve the models' prediction accuracy. The categorical values are then encoded with numbers so that machine learning models can use them to train themselves more effectively. Then, standardScaler is used to remove the features' high variation and different scales. It changes values feature by feature so that the mean of each characteristic becomes zero and the standard deviation becomes one. High variance characteristics have a negative impact on model performance. The principal component analysis (PCA) technique is then used to simplify the complexity of the data, eliminate noise, and discover latent features. Most significantly, it is used to minimize the dimensions of features and convert the values of correlated features in such a way that the characteristics become linearly uncorrelated. If any feature's correlation scores with other features are more than 0.3, the PCA method of translating features into uncorrelated features aids the model's performance. Four features in our sample have a correlation score greater than 0.3. As a result, PCA

is used to modify these linearly uncorrelated features.

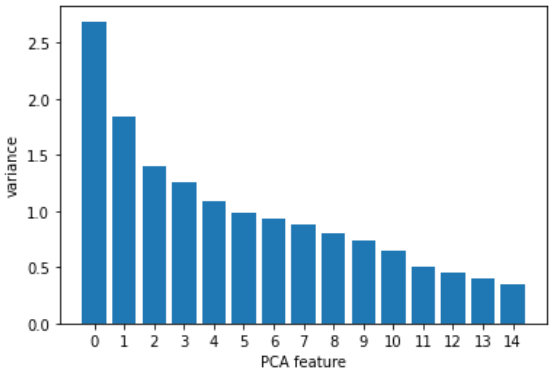


Figure 2: PCA implemented features' variance plot

Finally, the entire dataset is divided for model training and testing. 80% of the dataset is utilized for training, while the remaining 20% is used to evaluate the models' prediction performance on unseen data. Aside from this, one hot encoding on the label column is employed to train the deep neural network.

C. MODELS' PARAMETERS:

On the dataset, six different supervised machine learning algorithms are used. The GridSearchCV technique, which uses grid search cross validation, is used for figuring out the structures of these models. GridSearchCV is a hyperparameter tuning method that uses a grid to discover the models' ideal parameters after initializing a set of parameters. Five-fold cross-validation is used in the GridSearchCV. The complete dataset is used for this cross-validation.

K-Nearest Neighbor (KNN), Adaboost classifier (ADB), Support Vector Machine (SVM), Voting Classifier (VC), Random Forest classifier (RF), and Deep Neural Network (DNN) are the six supervised machine learning algorithms.

The GridSearchCV method uses the n_neighbors option for the KNN model. This parameter is used to determine the minimum number of points in the closest class that must be present for a new point to be considered a member of that particular class.

N_estimator and learning rate are utilized for grid searching cross validation for the AdaBoost classifier. The maximum number of estimators at which boosting terminates is N_estimator. These learners who are weak are also known as stumps and are estimators. The loss function used to determine the weight of the base models is controlled by learning rate.

The GridSearchCV method uses the Regularization Parameter C and kernel for the SVM model. The overall amount of regularization is negatively correlated with C's value. Using mathematical equations, the kernel parameters can alter the data such that a linear classifier can utilize it to solve a nonlinear problem.

A voting classifier is an algorithm for learning that predicts outcomes by combining the output from various base models or estimators. Three foundation models—KNN, SVM, and Decision tree—are employed in the implemented Voting Classifier model. The voting classifier's decision tree model is used to implement grid search. Cross validation for grid searching is done using the criterion variable. This option controls the impurity measurement technique for a split. The KNN and SVM basis models of the Voting Classifier employ the parameters' values from previously built grid searching cross-validation on KNN and SVM.

Grid searching cross-validation for the Random Forest classifier employs the n_estimator and criterion variables. The n-estimator parameter functions similarly to the Adaboost classifier's n-estimator, except that it does not use stumps. Because the Random Forest classifier makes use of weak learner trees, it is a strong learner. The criterion parameter functions similarly to the decision tree classifier's criteria parameter.

The parameters for employing the GridSearchCV technique are epochs for the deep neural network structure without batch input. This parameter determines how many times the model should train on the complete dataset. Additionally, batch_size and epochs are utilized as variables for implementing the GridSearchCV technique for the deep neural network model using batch input. How numerous samples from the dataset should be obtained all at once to train the model in each iteration is determined by the batch_size parameter.

The predefined parameters are applied instead of the GridSearchCV methods.

Table 1: GridSearchCV's best parameters' and default parameters' values of the implemented models

Models	Set of given parameters for GridSearch CV	GridSearchCV's Best Parameters	Default Parameters
KNN	n_neighbors =[5,10,15,20, 25]	n_neighbors= 15	weights='uniform', algorithm='auto', leaf_size=30, metric='minko

			wski', p=2
ADB	n_estimators = [100, 250, 500], learning_rate = [0.01, 0.1, 1.0]	n_estimators= 500, learning_rate= 1.0	algorithm='SAMME.R'
SVM	C=[50,20,10,1], kernel=['linear', 'poly', 'rbf', 'sigmoid']	C=50 kernel='rbf'	degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False
Voting Classifier (KNN, SVM, Decision Tree)	For Decision Tree, criterion=['gini', 'entropy', 'log_loss']	For Decision Tree, criterion= 'gini'	For Decision Tree, splitter='best', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, min_impurity_decrease=0.0, ccp_alpha=0.0
RF	n_estimator= [50,100,250, 500], criterion=['gini', 'entropy', 'log_loss']	N_estimator =50, criterion= 'gini'	min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='sqrt', min_impurity_decrease=0.0, bootstrap=True, oob_score=False,verbose=0, warm_start=False,ccp_alpha=0.0,
DNN without Batch input	epochs=[30,40,50,60,100]	epochs=30	activation='relu', activation='softmax', loss ='binary_crossentropy', optimizer='adam',metrics=['accuracy']
DNN with Batch input	epochs=[30,40,50,60,100], batch_size=[32,64,128,413]	epochs=30, batch_size=64	activation='relu', activation='softmax', loss= 'binary_crossentropy', optimizer='adam',metrics=['accuracy']

D. Deep Neural Network Structure

Layer (type)	Output Shape	Param #
dense_0 (Dense)	(None, 128)	2048
dense_1 (Dense)	(None, 64)	8256
dense_2 (Dense)	(None, 32)	2080
dense_3 (Dense)	(None, 2)	66

Total params: 12,450

Trainable params: 12,450

Non-trainable params: 0

A single input layer, three hidden layers, and one layer of output make up our deep neural network model as it is currently built. Due to the dataset's 15 training features, the input layer includes 15 neurons or perceptrons. In the first hidden layer, 128 neurons are utilized, followed by 64 neurons in the following hidden layer and 32 neurons in the third hidden layer. Due of the dataset's two distinct class labels, the output layer contains two neurons. The output layer used'softmax' as the activation function, whereas the hidden layers used'relu' as their activation function. The binary classification framework uses the 'binary_crossentropy' as a loss

function. The 'adam' optimizer stands for Adaptive Duration Estimation, a gradient descent optimization method. The 'gradient descent with momentum' algorithm and the 'RMSP' algorithm are combined to make it work.

III. EXPERIMENTAL RESULT AND DISCUSSION

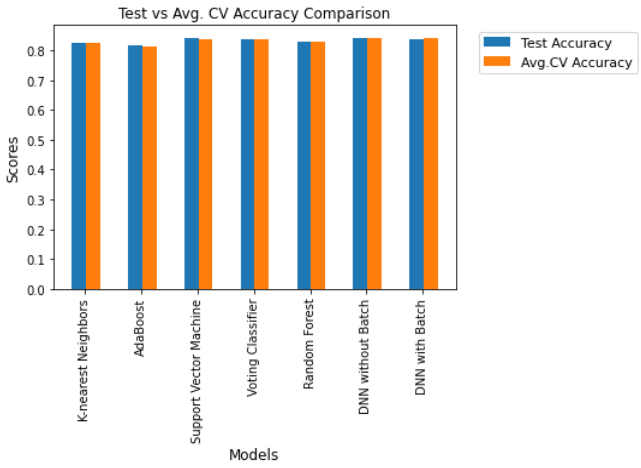


Figure 3: Models' Test Accuracy and Average Cross Validation Accuracy scores comparison plot

Table 2: Models' Test Accuracy scores and Average Cross Validation Accuracy scores comparison table:

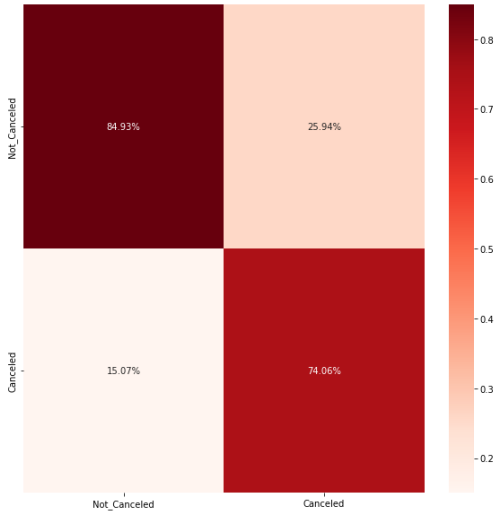
Models	**Test Accuracy Scores**	**Average Cross Validation Accuracy Scores**
KNN	0.82403	0.82378
ADB	0.81550	0.81172
SVM	0.84089	0.83835
VC	0.83604	0.83549
RF	0.83003	0.83087
DNN without batch input	0.84089	0.84080
DNN with batch input	0.83934	0.84320

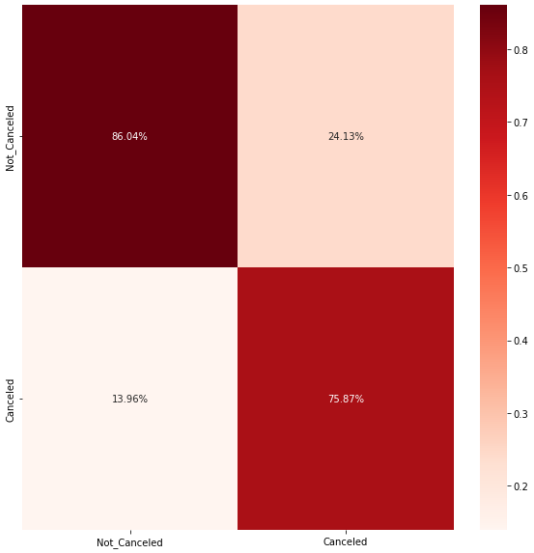
A deep neural network has the greatest degree of accuracy in both test results for accuracy and average cross-validation accuracy scores, as can be seen in the above table. The deep neural network model estimates the class labels more precisely because it employs a large number of parameters and weights and fine-tunes those weights during the training phase.
Compared to a deep neural network model trained without batch processing, the DNN model with batch processing has less testing accuracy and a greater cross-validation accuracy. Because the DNN with batch processing has been trained using 80% of the data in the entire dataset at the time of assessing testing accuracy. But the entire dataset is used when using cross-validation. In order to train and fine-tune its hyperparameters for every batch of input, the model received additional data in each batch. When batch processing is employed, DNN updates its weights after each batch, and when batch processing is not used, it updates its weights after each sample. As a result, DNN trained without batch processing outperformed DNN in the time frame of measuring testing accuracy on test data due to the lack of data in batch processing.

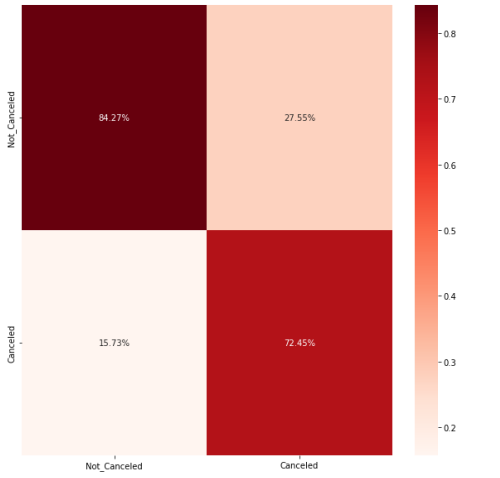
In terms of accuracy during testing and cross-validation, SVM is the second-best model. Because the regularization parameter is set to a high value in both instances, it has a higher accuracy score. The regularization parameter's inverse relationship with regularization's strength explains why having a high value for this parameter resulted in the SVM model suffering the least penalty. Regularization is performed to reduce overfitting in the model. Our data set is not excessively small, therefore the SVM model did not become overfit; rather, it had some issues with underfitting. Therefore, it has more closely matched the datasets after employing low regularization, and it has also done better in terms of prediction accuracy.

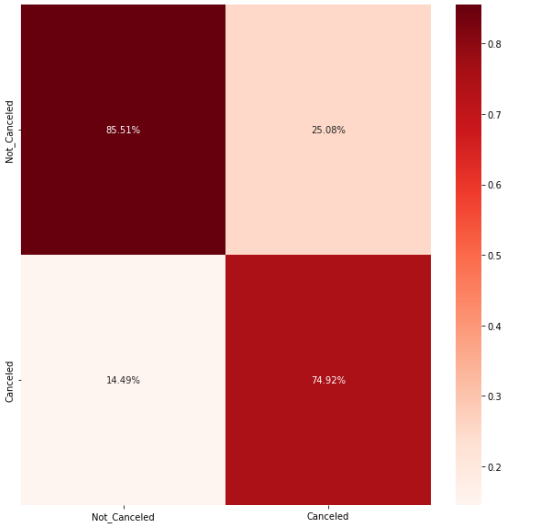
Among all the used models, Adaboost has provided the lowest accuracy ratings in terms of test precision and cross-validation accuracy. Since the AdaBoost classifier employs stumps, which have a single node and two leaves and are weak learners. Weak learners are also used as trees by the Random Forest classifier, although more than one features taken into account when creating those trees. AdaBoost only takes into account one feature while creating a stump, hence they are not very good at establishing precise classifications.
Below, you will see a heatmap of the confusion matrices for KNN, ADB, SVM, VC, RF, DNN

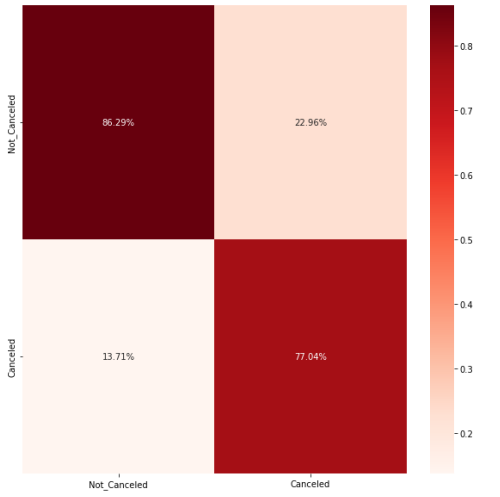
without batch processing, and DNN with batch processing, respectively.

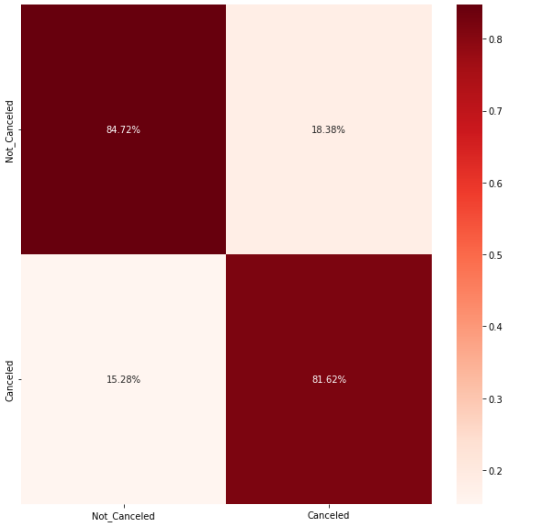


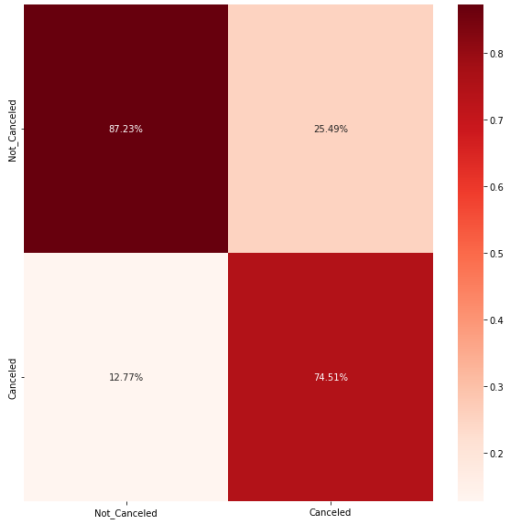












These heatmaps show that all the models predict the 'Not_Canceled' label more accurately than the 'Canceled' label. As a result of the dataset's 'Canceled' label having a far lower sample count then all other class labels. Because of this, models were unable to properly fit the parts of the "Canceled" label. As a consequence, the models made the 'Canceled' label predictions more inaccurately.

IV. CONCLUSION & FUTURE WORKS

The models require cautious fine-tuning of the model parameters with an extensive amount of data instances to have accurate classification model prediction. Additionally, not simply time but precision and precise classification are needed while developing the model for the algorithm.

In this study, six supervised classification models have been compared. Although in our example, we can observe that Deep Neural Networks models produce the best outcome. However, given that datasets are always incredibly different and unique, this cannot be true in all situations. Consequently, different categorization models may function better for certain datasets.

Only one type of dataset has been used in this study's performance comparison. In the future, comparative analyses can be made using several dataset formats to more accurately assess performance.

Colab Link of the code:

https://colab.research.google.com/drive/1W3eQcLI_j4tSCR5qHYZLdSfYoDPwJZds?usp=sharing