iJRASET
International Journal For Research in Applied Science and Engineering Technology

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 5 **Issue:** XII **Month of publication:** December 2017

**DOI:**

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor :6.887*
*Volume 5 Issue XII December 2017- Available at www.ijraset.com*

# Predictive Analysis of Online Auctions Using XGBoost

Sandeep Kumar[1]
[1]*Computer Science & Engineering Department, MSIT, GGSIPU.*

***Abstract: In this paper, the dataset used is analysed for making the price prediction of online auctions. Among supervised machine learning models, one of the most promising model is Extreme Gradient Boosting Model. Here the dataset parameter tuning is shown The corresponding density plots on train and test data are used to display the results. The price predictability is given in terms of the price bucket and the approximate price for an online auction. As a conclusive remark the comparison of XGBoost with other machine learning models is given at the end.***
***Keywords: XGBoost, supervised learning, online auctions, price prediction.***

## I. INTRODUCTION

Machine learning models are widely used for carry out the data analysis. These help in a confident way to make the predictions. The Extreme Gradient Boosting Machine Learning Model is used for the supervised learning class of the problems. This is based on the "Greedy Function Approximation: A Gradient Boosting Machine" work by Friedman [1]. Though boosting is not a new method. In this model train and test method is used for making the predictions [4, 5]. These predictions further can be utilized in either way i.e. one can use these for performing the regression task or the classification task. For example it can be used to find the belongingness of predicted price element to the positive class and it can be used to provide the weightage to that predicted element, to support it. Next important step is parameter tuning, in this unknown elements of datasets are tried to present. After parameter tuning the objective function is decided. The output according to this objective function determines the accuracy. Here the output function is optimized for better results.

## II. IMPLEMENTATION AND ANALYSIS

### *A. XgBoost Classification of The Data Set*

XgBoost algorithm is one of the best model in data science today. It is much better as compared to the traditional Random Forest or Neural Network models. In this paper it will be shown by comparing the model with other ones. Here the reading of dataset in data. table format is taken because it is faster. Then variables are removed, that has unique values or are not needed in analysis.

*Code Block 1. Reading the used Data set[5].

```
train <- fread("C://auction/Raw Data set/Raw Data set/TrainingSet.csv")
Read 65.7% of 258588 rows
Read 258588 rows and 28 (of 28) columns from 0.046 GB file in 00:00:03
test <- fread("C://auction/Raw Data set/Raw Data set/TestSet.csv")
train <- train[, c("EbayID", "SellerName", "EndDay" ) := NULL]
test  <- test[, c("EbayID", "SellerName", "EndDay") := NULL]
```

Then we extract explanatory variable for train and test dataset and create xgboost.matrix objects. Xgboost requires only numerical values, in matrix form. So first, we need to extract explanatory variable and then melt it in xgb.D Matrix object.

*Code Block 2 Explanatory variable for train and test dataset.

```
outcome.train <- train[, QuantitySold]
outcome.test  <- test[, QuantitySold]
# Create XgBoost objects
xgb.train <- xgb.DMatrix(data.matrix(train[, !"QuantitySold"]), label = outcome.train)
xgb.test  <- xgb.DMatrix(data.matrix(test[, !"QuantitySold"]), label = outcome.test)
```

Machine learning methods require to tune parameters. First step in analysis is to create an object, then tune some parameters that decide how well our model will be [2]. For xgboost model, we have set of parameters shown below:

*1)*eta: show how "long" it takes to take one step (iteration) in algorithm

*2)* max_depth - as xgboost model is tree based model; it can specify how deep this tree can be. Of course, deeper the tree is than the better accuracy, but only on training dataset. Too deep tree can provide to over fitting, which cause poor accuracy on test data set.

*3)*sub sample - how many observation is taken to one iteration

*4)*colsample_bytree - amount of variable that is randomly taken at every step

*5)*eval_metric <- this indicates metric that we want to base on this model. In this situation, as we deal with classification/segmentation model, decide to use receiver operator characteristic curve, which is very simple in interpretation: higher ROC values (boundary 0-1) = higher accuracy of prediction. Next step is to create a model. Here we set argument maximize = TRUE, because we want our measure (eval_metric) to be as high as possible on test data set.

Table 1. Parameters for Extreme Gradient Boosting.

```
# List of parameters
param <- list( objective      = "binary:logistic",
eta           = 0.1,
max_depth     = 8,
subsample     = 1,
colsample_bytree = 0.6,
eval_metric     = "auc")
Training the XgBoost Model.
xgb_mod <- xgb.train( params = param,
data = xgb.train, nrounds = 50,
verbose = TRUE,
maximize = TRUE)
```

Section below shows an importance case. This tables (plots) show us how many information each of variable gives to the explanatory variable. It is a very informative task if we have thousands of variables.

Code Block 3 XgBoost Importance Case.

```
xgb_imp <- xgb.importance(colnames(test[, !"QuantitySold"]), model = xgb_mod)
xgb.plot.importance(xgb_imp)
```

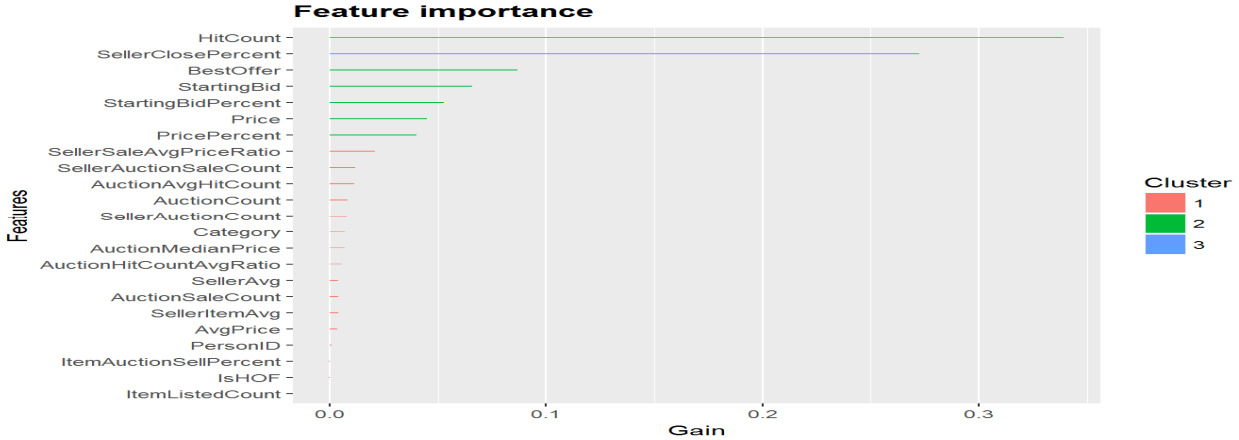



Fig. 1. Feature importance in dataset.

*B. Confusion Matrix for train data set*

Confusion Matrix gives information about overall accuracy of the model, and also a clue in this analysis i.e. what is the rate of prediction good case [6]? It is desirable to maximize the probability of recognizing good cases and minimize error of prediction bad cases as good cases (it is called false-positive rate). The confusion matrix are calculated for both data set to see if there is big difference between two samples. If differences are big, this is a first signal of over fitting [3].

Table 2. Confusion Matrix for Training Data set.

```
xgb.train.pred <- predict (xgb_mod, newdata = xgb.train)
xgb.train.pred.bin <- ifelse (xgb.train.pred < 0.5, 0, 1)
conf_train <- confusionMatrix (xgb.train.pred.bin, outcome.train)
conf_train
## Confusion Matrix and Statistics
## Reference
## Prediction     0     1
## 0 172558 17212
## 1   6275 62543
## Accuracy: 0.9092
## 95% CI: (0.9081, 0.9103)
## No Information Rate: 0.6916
## P-Value [Acc > NIR] : < 2.2e-16
## Kappa: 0.7787
## Mcnemar's Test P-Value: < 2.2e-16
## Sensitivity: 0.9649
## Specificity: 0.7842
## Pos Pred Value: 0.9093
## Neg Pred Value: 0.9088
## Prevalence: 0.6916
## Detection Rate: 0.6673
## Detection Prevalence: 0.7339
## Balanced Accuracy: 0.8746
## 'Positive' Class: 0
```

Table 3. Confusion Matrix for Test Dataset.

```
xgb.test.pred <- predict (xgb_mod, newdata = xgb.test)
xgb.test.pred.bin <- ifelse (xgb.test.pred <0.5, 0, 1)
conf_test <- confusionMatrix (xgb.test.pred.bin, outcome.test)
conf_test
## Confusion Matrix and Statistics
## Reference
## Prediction     0     1
##   0 27304 2620
##   1   757 6779
##    Accuracy: 0.9099
##    95% CI: (0.9069, 0.9127)
##    No Information Rate: 0.7491
##    P-Value [Acc > NIR] : < 2.2e-16
##    Kappa: 0.7433
```

```
##   Mcnemar's Test P-Value: < 2.2e-16
##   Sensitivity: 0.9730
##   Specificity: 0.7212
##   Pos Pred Value: 0.9124
##   Neg Pred Value: 0.8995
##   Prevalence: 0.7491
##   Detection Rate: 0.7289
##   Detection Prevalence: 0.7988
##    Balanced Accuracy: 0.8471
##   'Positive' Class: 0
```

*C. Density plots for XgBoost*

Density Plots are very informative and can show us where to set Cutoff point. Cutoff point can be understood as a point which is used to make a difference between two score points. In our case, it is very clear, as our result is interpretable in probability sense. If observation takes the result higher than 0.5 than we have very high probability that this case will be recognized by model as good cause, and it will be true. And otherwise, probability lower than 0.5 indicates higher chance that our observation will not sell a product.

*1) Density Plot for Train Dataset:*

*Code Block 4. Code for Density Plot for Training Dataset.

```
xgb.train.pred <- data.frame(xgb.train.pred)
dens_plot_train <- ggplot(xgb.train.pred, aes(x= xgb.train.pred, colour = factor(xgb.train.pred.bin),
fill = factor(xgb.train.pred.bin), alpha = 0.3)) +
geom_density()
ggplotly(dens_plot_train)
```

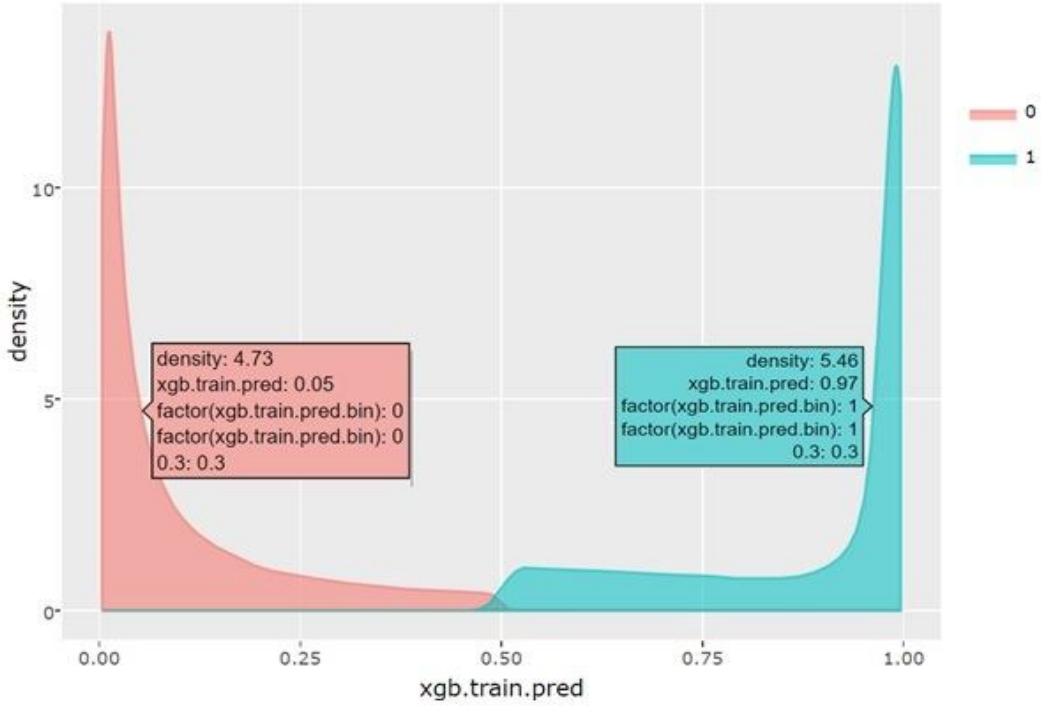



Fig. 2. Density plot for train dataset

*2) Density Plot for Test Dataset*

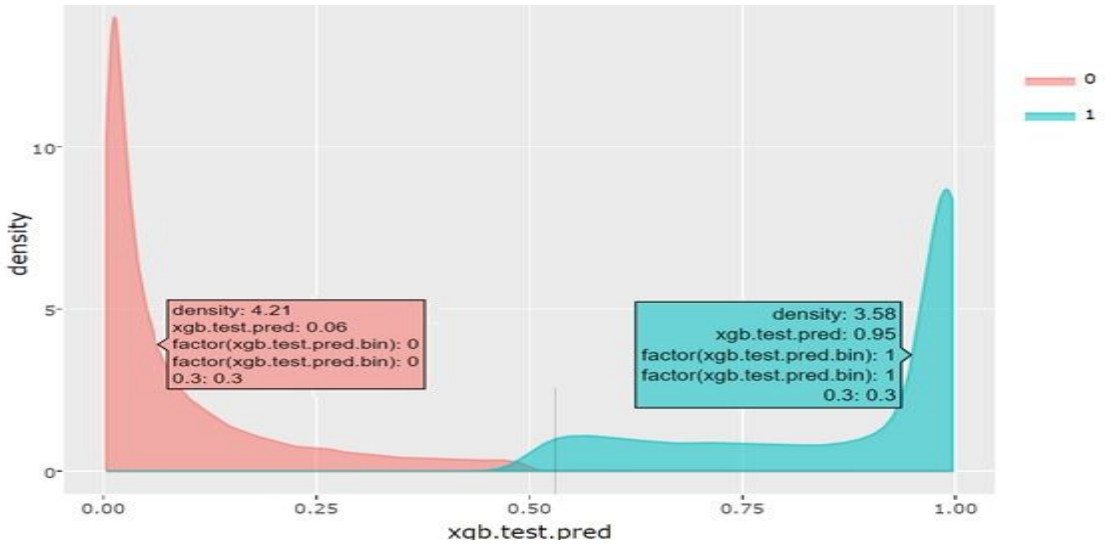



Fig. 3. Density plot for test dataset-III.

## III.COMPARING THE PERFORMANCE

Table 4. Accuracy Result of Extreme Gradient Boosting.

```
xgb_train_1
## eXtreme Gradient Boosting
## 258588 samples
## 9 predictor
## No pre-processing
## Resampling: Cross-Validated (3 fold)
## Summary of sample sizes: 172392, 172393, 172391
## Resampling results:
##   RMSE      Rsquared
##   51.89078 0.9099868
## Tuning parameter 'nrounds' was held constant at a value of 2
## 0.8
## Tuning parameter 'min_child_weight' was held constant at a value of
## 1
## Tuning parameter 'subsample' was held constant at a value of 0.6
```

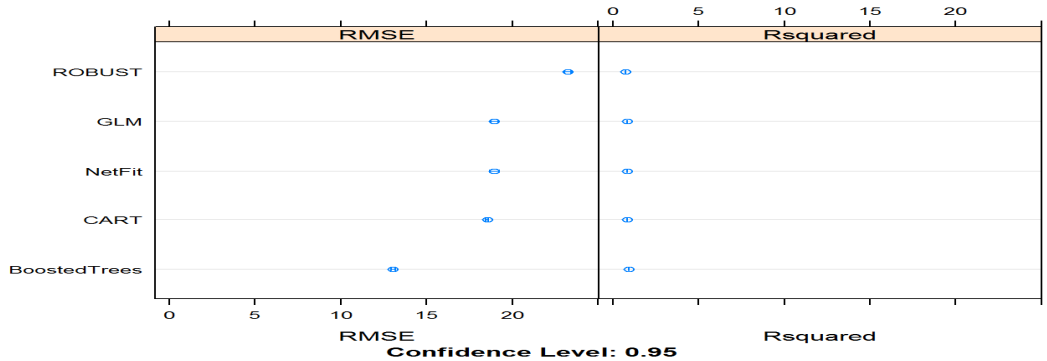



Fig. 4. Efficiency comparison of models.

The figure above shows a comparison of extreme gradient boosting with other machine learning models. It is quite evident that the model seems to perform better.

## IV. CONCLUSIONS

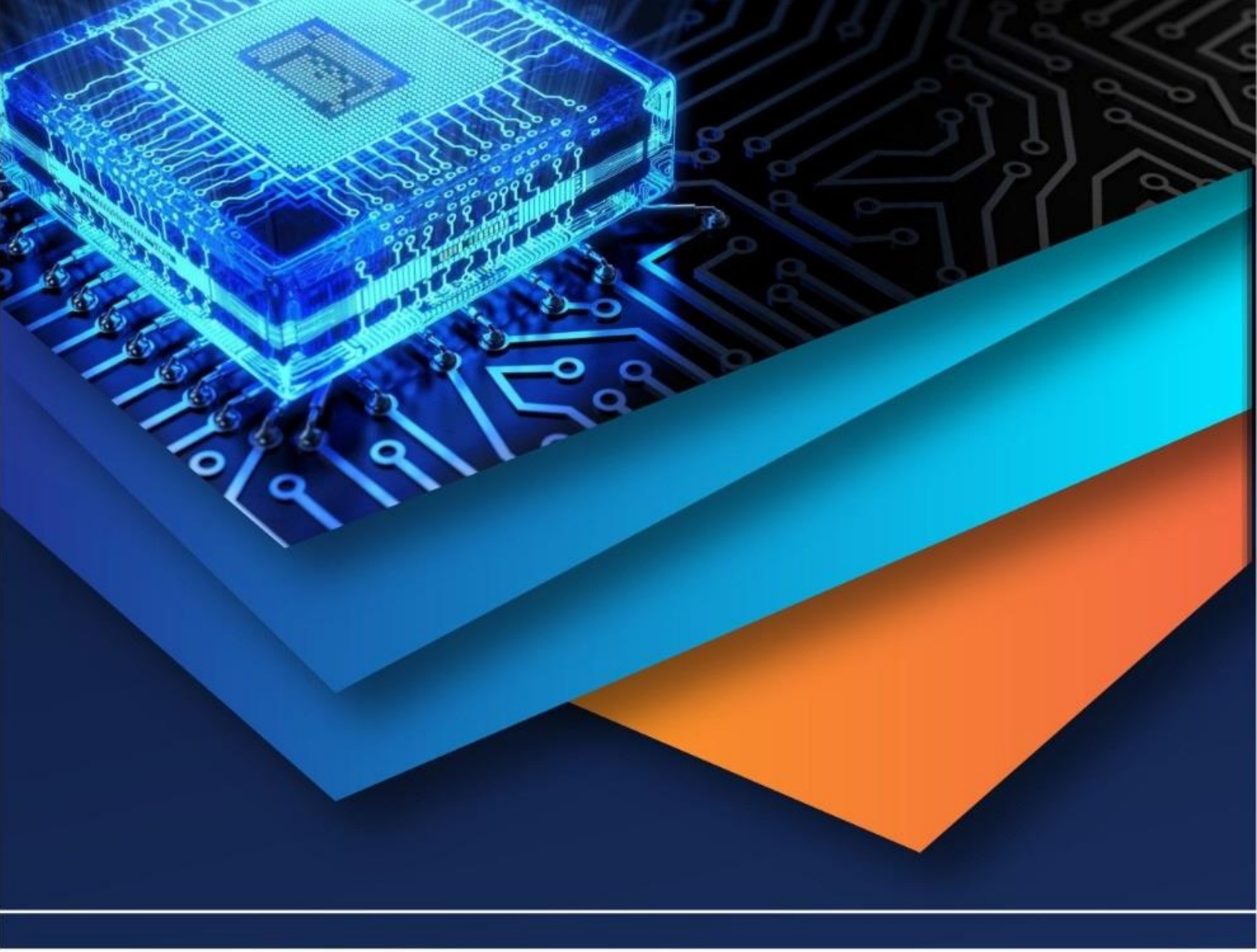
From the above discussion and result interpretation it is quite clear that the XGBoost gives very promising results. The dataset was specifically taken about the online auctions. The values of Root Mean Square Error is 51.89078 add the Rsquared is 0.9099868. The results tends to further improve by using the other available datasets and improvements in the area. The work can be further expanded to the similar other predictive areas.

## V. ACKNOWLEDGMENT

I would like to thank Dr. Rahul Rishi sincerely. His motivation, positivity and direction always guides me to keep my work going.

## REFERENCES

[1] J. F.-A. of statistics and undefined 2001, “Greedy function approximation: a gradient boosting machine,” JSTOR.
[2] 101.T. Likhomanenko and A. Rogozhnikov, “Improving Reproducibility of Science Irreproducibility Indicators,” no. July, 2015.
[3] R. Iniesta, D. Stahl, and P. McGuffin, “Machine learning, statistical learning and the future of biological research in psychiatry,” Psychol. Med., 2016.
[4] S. Kumar and R. Rishi, “Hybrid dynamic price prediction model in online auctions,” Int. J. Appl. Eng. Res., vol. 12, no. 5, pp. 598–604, 2017.
[5] Grossman Jay, “Jay Grossman.” [Online]. Available: http://jaygrossman.com.
[6] M. Miles, A. Huberman, J. S.-B. Hills, and undefined 1984, “Qualitative data analysis: A sourcebook,” JSTOR.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY