Assignment 2

BUAN 6341.501 – Applied Machine Learning
Done by: Weiyang Sun

Learning Algorithms Covered

1. Support Vector Machines
    a. Linear Kernel
    b. Polynomial Kernel
    c. Radial Basis Function (RBF) Kernel
2. Decision Trees
3. Random Forest
4. Gradient Boosted Classifier

Datasets Used

- Dataset 1: Student Performance Dataset (Math Scores)
    - Binary Classification: 1 = Above Median Score ; 0 = Below Median Score
- Dataset 2: Telecommunication Churn Dataset*
    - Binary Classification: 1 = Churn ; 0 = No Churn

*Random sampling from Dataset 2 was performed to ensure the same amount of data (395 rows) as Dataset 1. Telecom Churn was chosen because it utilizes the similar concept of predicting a binary outcome using a set of individual's characteristics, like that of student's performance. Since telecom churn is a widely researched field, it would provide us with a good platform to test our intuitive understanding of classification models before extending this to other fields.

In all the following experimentations, cross-validation has been applied. This would help us to better estimate (by providing us with a confidence interval – denoted by the various shadings in the various graphs) the ability of the algorithm to perform on an unseen data. This is in hope of generating a less biased or less optimistic estimate as compared to a simple train/test split.

Results Summary

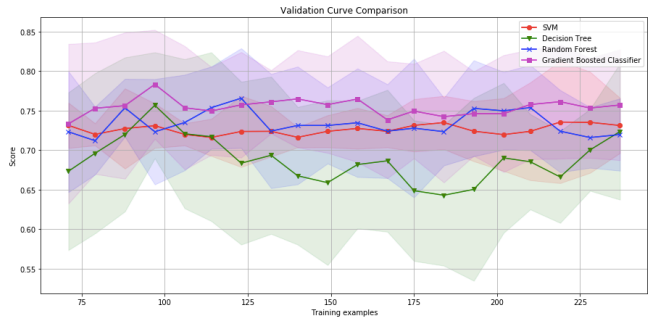
The table below shows the overall accuracy:

The columns in **GREEN** represents the model after hyperparameter tuning. The columns in **RED** represents the model before hyperparameter tuning.

	Student Performance	Student Performance	Telecom Churn	Telecom Churn
Support Vector Machines (Linear)	65.0%	65.6%	79.0%	78.6%
Support Vector Machines (Polynomial)	56.0%	--	74.0%	--
Support Vector Machines (RBF)	60.0%	--	79.0%	--
Decision Tree	60.0%	64.1%	75.0%	79.3%
Random Forest	56.0%	69.4%	79.0%	78.6%
Gradient Boosted Classifier	66.0%	64.1%	78.0%	76.3%

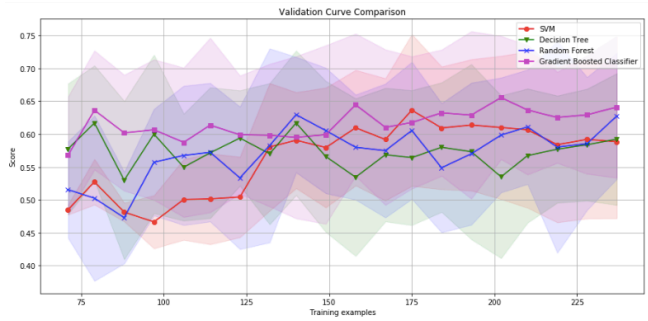
In some instances, after hyperparameter tuning, the accuracy of the model decreased. This could be because, the same standardized search grid was adopted for both datasets. Therefore, there could be instances where the optimal configuration for certain models were not included. A randomized search grid could have been a better first option before a standardized search grid is deployed.

These are the various learning curves for the *telecom churn*:



As training examples increases, the model recommended is the Gradient Boosted Classifier.

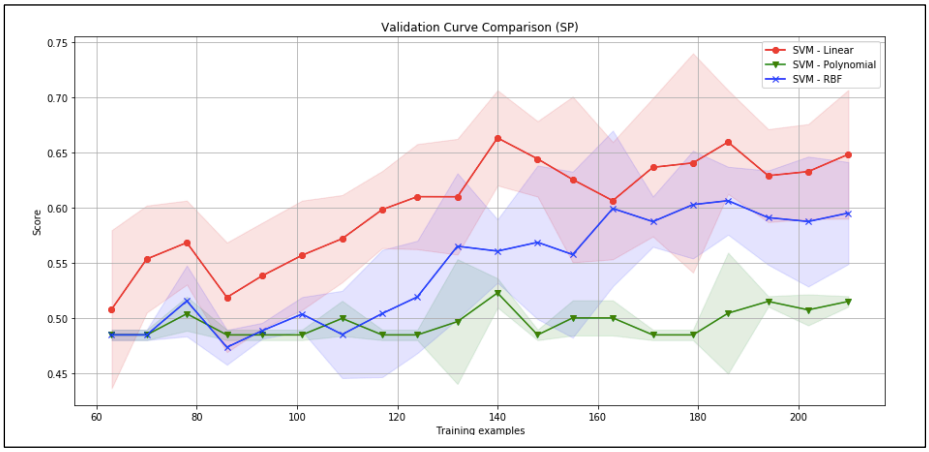
These are the various learning curves for the *student performance*:

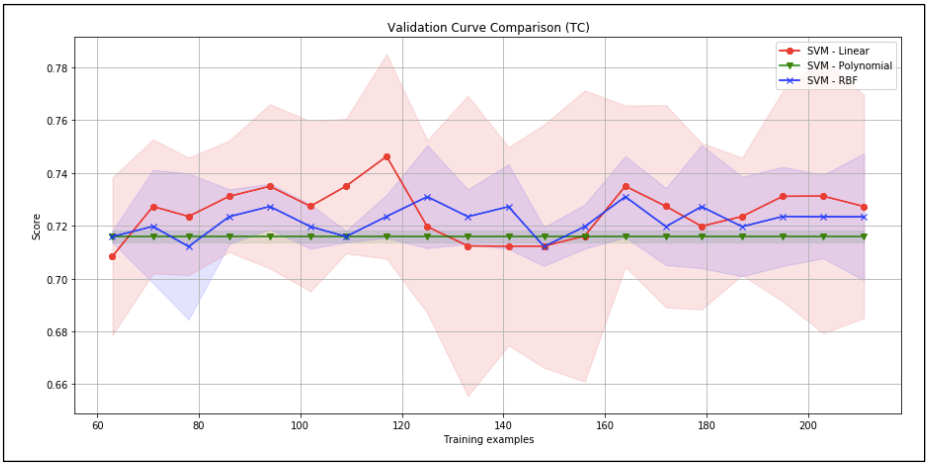


As training examples increases, the model recommended is the Gradient Boosted Classifier.

Support Vector Machines

Plotting out the learning curves showing the cost against training examples, for the different types of kernel (top side – Student Performance (SP) ; bottom side – Telecom Churn (TC)):

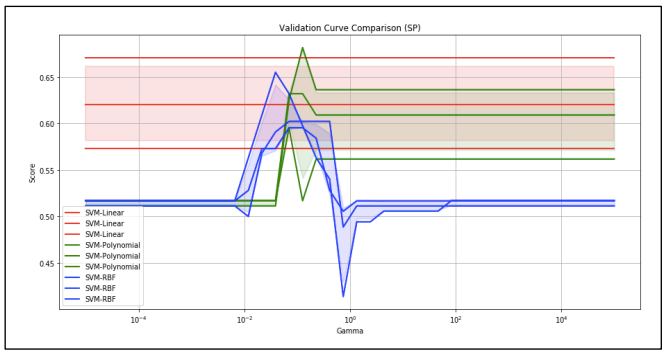


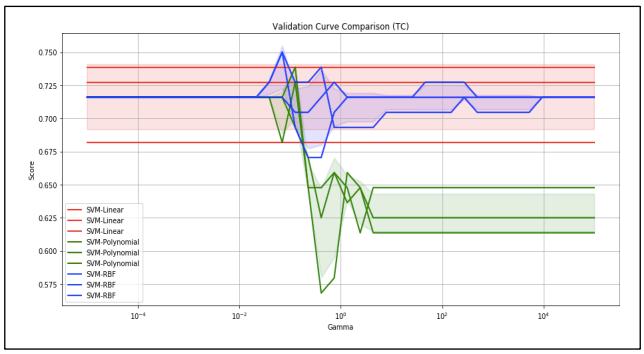


We noticed that there are instances of numerical instabilities present in using other kernels besides linear, however, in general, test score's cost either remains the same or increases with an increase training examples – combat high variances.

If a non-linear kernel function is used, the smoothness of the kernel function has an effect on the complexity of the classifier and risk over-fitting. A Radial Basis Function (RBF) kernel, for example, with a small-scale factor (kernel parameter) will tend towards a Linear kernel. If a high scale factor is used, the output of the classifier will be very sensitive to small changes in the input, leading to over-fitting. The performance of an SVM can be sensitive to the selection of the regularization and kernel parameters, and it is possible to get over-fitting in tuning these hyper-parameters.

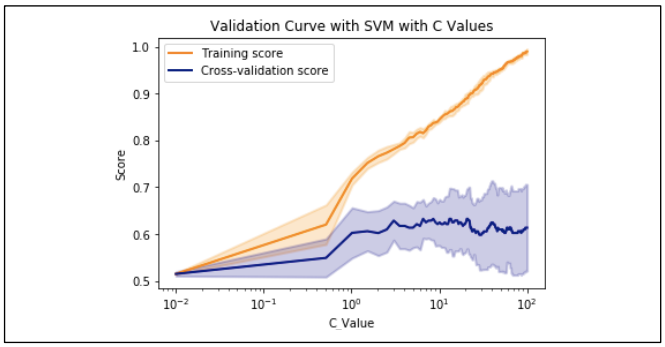
Plotting out the learning curves showing the cross-validation scores (cv=3) against gamma, for the different types of kernel (left side – Student Performance (SP) ; right side – Telecom Churn (TC)):

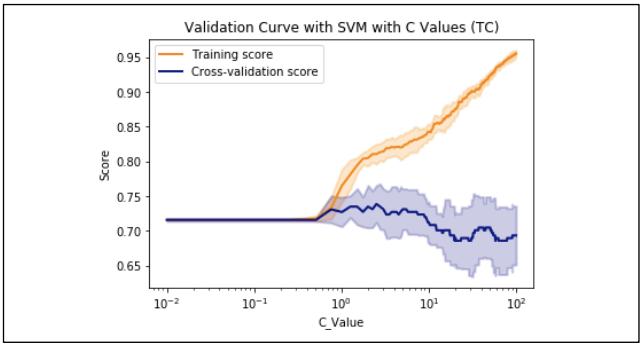




Gamma is a parameter specifically for nonlinear support vector machines, in this case, the polynomial and radial basis function kernels. This would explain why SVM with linear kernels are unaffected by a change in gamma.

A small gamma placed on an SVM causes larger variance (influence by x_j is more) implies the class of this support vector will have influence on deciding the class of the vector x_i even if the distance between them is large. If gamma is large, then variance is small implying the support vector does not have wide-spread influence. Therefore, a large gamma leads to high bias and low variance models, and vice-versa. By plotting out this learning curve, we can find the bias-variance trade-off point.



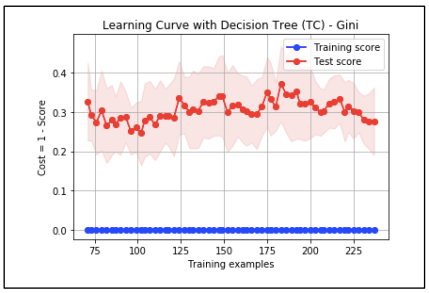


The goal of SVM is to find a hyperplane that would leave the widest possible "cushion" between input points from two classes. There is a tradeoff between "narrow cushion, little / no mistakes" and "wide cushion, quite a few mistakes". Often times it is desirable to allow some training points to be misclassified in order to have a generalized model.

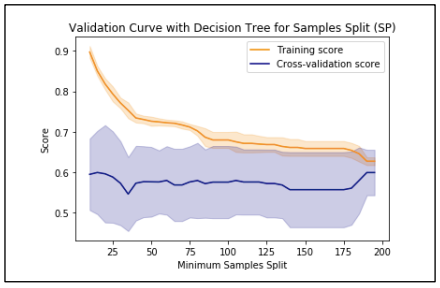
C is the parameter for the margin cost function, which controls the influence of each individual support vector. A large C makes the cost of misclassification high - giving us low bias and high variance and a narrower cushion. A small C makes the cost of misclassification low - giving us higher bias and lower variance and a wider cushion. Plotting the learning curve allows us to pick the trade-off point.

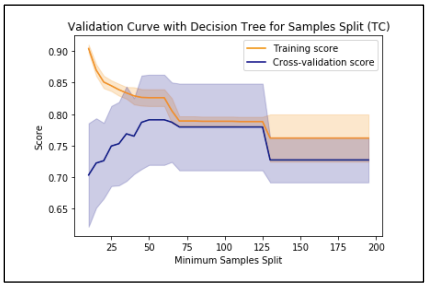
Decision Trees

Plotting out the learning curves showing the cost against training examples for Gini Impurity (left side – Student Performance (SP) ; right side – Telecom Churn (TC)):

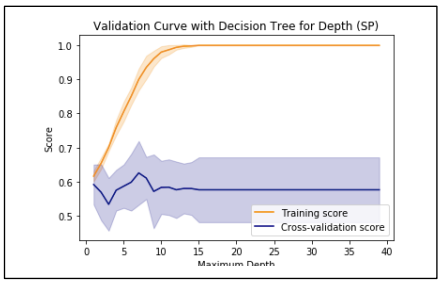


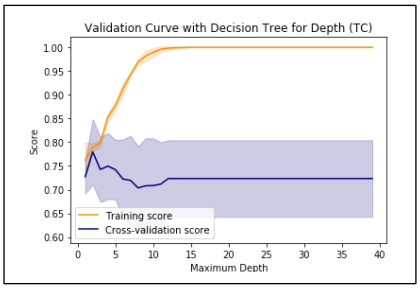
After reading the research paper titled 'Theoretical comparison between the Gini index and information gain criteria', it can be proven that Gini Impurity and Information Gain Entropy are similar. It was shown that in only 2% of cases, would this decision significantly matter. Entropy is slower computational because of the logarithmic functions, whereas Gini Impurity does not utilize this, allowing for a closed form solution.





The parameter minimum samples split specifies the minimum number of samples required to split an internal node. Therefore, the smaller the minimum number of samples splits leads to high variance and low bias (over-fitting) vice-versa. This behavior can be observed from the validation curve plotted above.

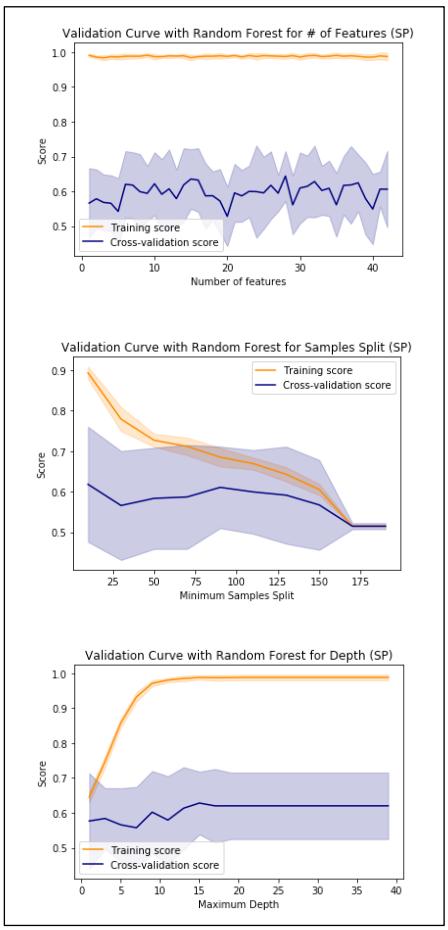


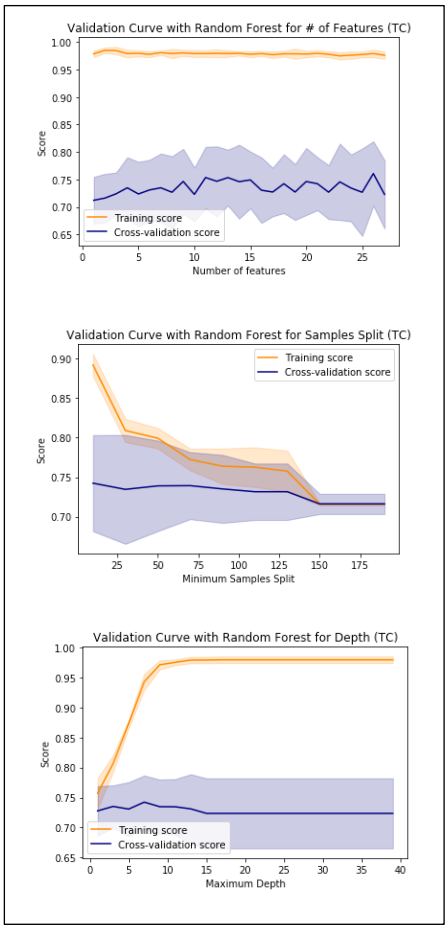


The parameter maximum depth controls the depth of the decision tree to be created. It can be described as the length of the longest path from the tree root to a leaf. This means that the root node is considered to have a depth of 0. Therefore, a longer path leads to over-fitting (higher training scores) but does not generalized well in terms of cross-validation score, as seen in the learning curve above.

Random Forest

Similar to decision trees, the choice of whether to use Gini Impurity or Information Gain Entropy does not make a significant difference. For the associated learning curves, please refer to the Jupyter Notebook.





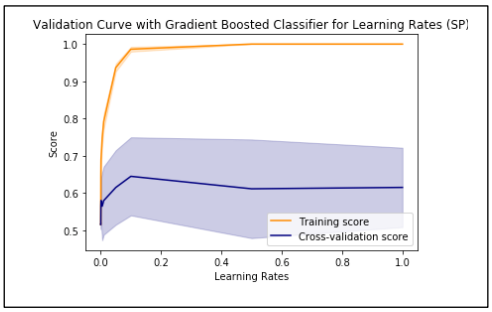
Random Forest algorithm is a supervised classification algorithm. It creates a forest in a random fashion. There is a direct relationship between the number of trees in the forest and the results it can get: the larger the number of trees (n estimators), the more accurate the result. The difference between Random Forest and decision trees lies in the processes of finding the root nodes and splitting the feature nodes at random.

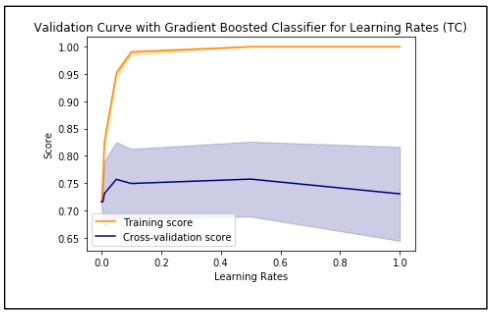
However, besides this randomness in finding root nodes and splitting, the learning curves from maximum features, minimum samples split, maximum depth, maximum leaf nodes and minimum leaf samples are identical for a random forest and decision trees. The same explanation used for decision trees can be applied into understanding random forest learning curves.

Gradient Boosting Classifier

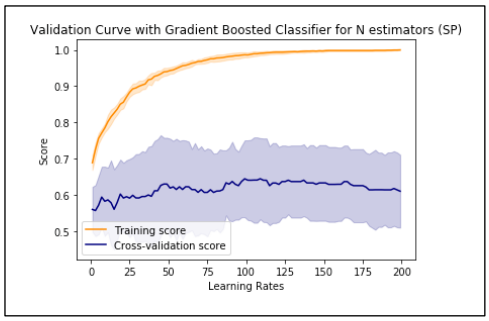
Gradient Boosting using scikit-learn builds an additive model in a forward stage-wise fashion. This allows for the optimization of arbitrary differentiable loss functions. In each stage the regression trees are fit on the negative gradient of the binomial deviance loss function.

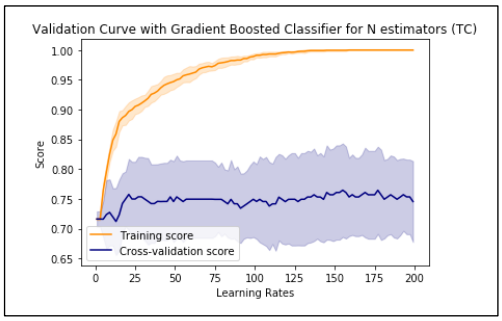
In our experiment, we begin first by changing learning rates. This determines the impact of the regression trees on the final outcome. Lower values are preferred as it makes the model robust to the specific characteristics of tree and thus allowing it to generalize well. However, it would require higher number of trees to model all the relations and will be computationally expensive. This can be seen through the following learning curves:





N estimators refers to the number of boosting stages (weak learners) to perform. Gradient boosting is fairly robust to over-fitting; therefore, a large number of boosting stages usually results in better performance. Since learning rate determines the impact of a weak learner on the final outcome, there is a trade-off between learning rate and n estimators.





Even though in general, gradient boosting is fairly robust to over-fitting, there is a threshold in which that does not hold true. By plotting the learning curve, we can see that increasing the number of estimators without any limits can result in overfitting also (high training scores, low cross-validation scores). In our case, using about 25-30 trees is optimal.

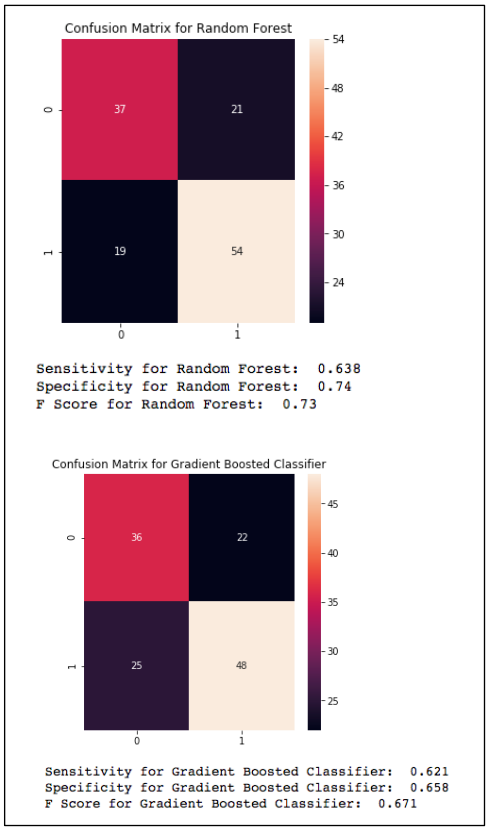
Performance Comparison

As seen in the results summary, Gradient Boosting Classifier yielded the best accuracy. However, we have to give due consideration to the Sensitivity and Specificity of the model.

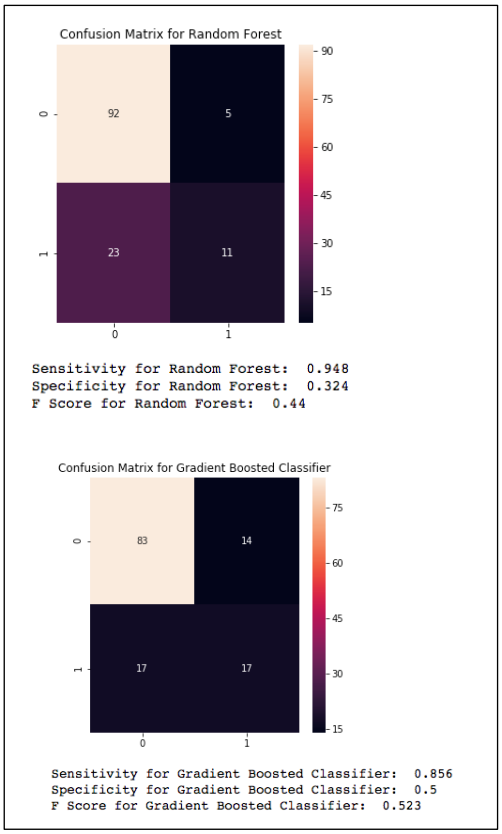
In the case of Student Performance: Type 1 error is more important. This is because if a student is below the median score, and we predict that he/she is above, little remedial action can be done to help. (High Specificity)

In the case of Telecom Churn: Type 2 error is more important (where the customer is churning, but the model predicts no churn). This would result in a lack of action and a potential loss in revenue. (High Sensitivity)

STUDENT PERFORMANCE



TELCOM CHURN



Taking into consideration, accuracy, F-score and the respective errors which are important, we recommend Random Forest (after hyper-parameter tuning) for both the student performance and telecom churn dataset. This would give us the highest specificity/sensitivity with the highest accuracy. However, note that the F-score (which serves to balance precision/recall) is not as high as the gradient boosted classifier.