

x

-

(<http://play.google.com/store/apps/details?id=com.analyticsvidhya.android>)



[LOGIN / REGISTER \(HTTPS://ID.ANALYTICSVIDHYA.COM/ACCOUNTS/LOGIN/?](https://id.analyticsvidhya.com/accounts/login/?)

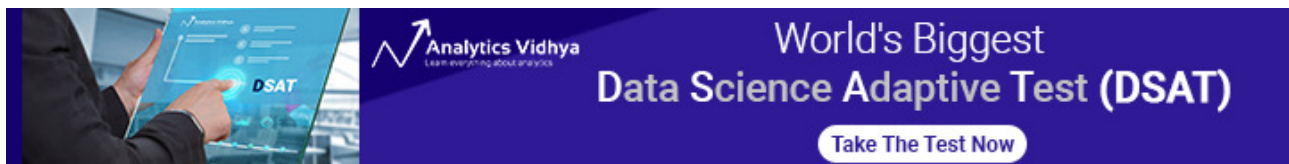
[NEXT=HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2018/05/IMPROVE-MODEL-PERFORMANCE-CROSS-VALIDATION-IN-](https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/)

[PYTHON-R/](https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/)



(<https://www.analyticsvidhya.com/myfeed/?>

[utm-source=blog&utm-medium=top-icon/](https://www.analyticsvidhya.com/myfeed/?utm-source=blog&utm-medium=top-icon/))



([https://dsat.analyticsvidhya.com/?utm\\_source=blog&utm\\_medium=top-right](https://dsat.analyticsvidhya.com/?utm_source=blog&utm_medium=top-right))

[BEGINNER \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/BEGINNER/\)](https://www.analyticsvidhya.com/blog/category/beginner/)

[BUSINESS ANALYTICS \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/BUSINESS-ANALYTICS/\)](https://www.analyticsvidhya.com/blog/category/business-analytics/)

[CLASSIFICATION \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/CLASSIFICATION/\)](https://www.analyticsvidhya.com/blog/category/classification/)

[MACHINE LEARNING \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/MACHINE-LEARNING/\)](https://www.analyticsvidhya.com/blog/category/machine-learning/)

[PYTHON \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/PYTHON-2/\)](https://www.analyticsvidhya.com/blog/category/python-2/)

[R \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/R/\)](https://www.analyticsvidhya.com/blog/category/r/)

[SUPERVISED \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/SUPERVISED/\)](https://www.analyticsvidhya.com/blog/category/supervised/)

[TECHNIQUE \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/TECHNIQUE/\)](https://www.analyticsvidhya.com/blog/category/technique/)

## Improve Your Model Performance using Cross Validation (in Python and R)

[SUNIL RAY \(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/AUTHOR/SUNIL-RAY/\)](https://www.analyticsvidhya.com/blog/author/sunil-ray/), MAY 3, 2018 [LOGIN TO BOOKMAR...](#)

*This article was originally published on November 18, 2015, and updated on April 30, 2018.*

## Introduction

One of the most interesting and challenging things about data science ([https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm\\_source=blog&utm\\_medium=ImproveModelPerformanceCrossValidationarticle](https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm_source=blog&utm_medium=ImproveModelPerformanceCrossValidationarticle)) hackathons is getting a high score on both public and private leaderboards. I have closely monitored the series of data science hackathons ([https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm\\_source=blog&utm\\_medium=ImproveModelPerformanceCrossValidationarticle](https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm_source=blog&utm_medium=ImproveModelPerformanceCrossValidationarticle)) and found an interesting trend. This trend is based on participant rankings on the public and private leaderboards.

One thing that stood out was that participants who rank higher on the public leaderboard lose their position after their ranks gets validated on the private leaderboard. Some even failed to secure rank in the top 20s on the private leaderboard (image below).

Eventually, I discovered the phenomenon which brings such ripples on the leaderboard.

Name	Public Leaderboard Rank	Private Leaderboard Rank
Alps	1	42
Isac	2	33
Piero	3	2
Zak	4	21
Mic	5	50
Harv	6	35
Mike	7	1
Mine	8	9
Root	9	40
Flint	10	48

(<https://www.analyticsvidhya.com/wp-content/uploads/2015/11/rank.png>)

Take a guess! What could be the possible reason for high variation in these ranks? In other words, why does their model lose stability when evaluated on the private leaderboard?

In this article, we will look at possible reasons for this. We will also look at the concept of cross validation and a few common methods to perform it.

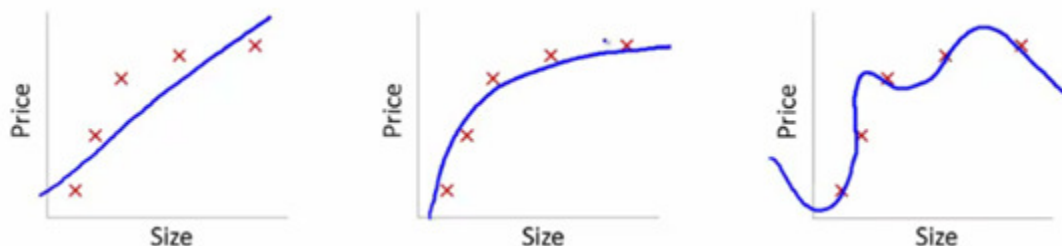
**Note:** This article is meant for every aspiring data scientist ([https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm\\_source=blog&utm\\_medium=ImproveModelPerformanceCrossValidationarticle](https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm_source=blog&utm_medium=ImproveModelPerformanceCrossValidationarticle)) keen to improve his/her performance in data science competitions. Each technique is followed by code snippets from both R and Python ([https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm\\_source=blog&utm\\_medium=ImproveModelPerformanceCrossValidationarticle](https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm_source=blog&utm_medium=ImproveModelPerformanceCrossValidationarticle)).

## Table of Contents

1. Why do models lose stability?
2. What is cross validation?
3. A few common methods used for cross validation
  - The Validation set Approach
  - Leave out one cross validation (LOOCV)
  - k-fold cross validation
  - Stratified k-fold cross validation
  - Adversarial validation
  - Cross validation for time series
  - Custom cross validation techniques
4. How to measure the model's bias-variance?

## Why do models lose stability?

Let's understand this using the below snapshot illustrating the fit of various models:



<https://www.analyticsvidhya.com/wp-content/uploads/2015/11/15.png>

Here, we are trying to find the relationship between size and price. To achieve this, we have taken the following steps:

1. We've established the relationship using a linear equation for which the plots have been shown. The first plot has a high error from training data points. Therefore, this will not perform well on either public or the private leaderboard. This is an example of **"Underfitting"**. In this case, our model fails to capture the underlying trend of the data
2. In the second plot, we just found the right relationship between price and size, i.e., low training error and generalization of the relationship
3. In the third plot, we found a relationship which has almost zero training error. This is because the relationship is developed by considering each deviation in the data point (including noise), i.e., the model is too sensitive and captures random patterns which are present only in the current dataset. This is an example of **"Overfitting"**. In this relationship, there could be a high deviation between the public and private leaderboards

A common practice in data science competitions is to iterate over various models to find a better performing model. However, it becomes difficult to distinguish whether this improvement in score is coming because we are capturing the relationship better, or we are just over-fitting the data. To find the right answer for this question, we use validation techniques. This method helps us in achieving more generalized relationships.

## What is Cross Validation?

Cross Validation is a technique which involves reserving a particular sample of a dataset on which you do not train the model. Later, you test your model on this sample before finalizing it.

Here are the steps involved in cross validation:

1. You *reserve* a sample data set
2. Train the model using the remaining part of the dataset
3. Use the reserve sample of the test (validation) set. This will help you in gauging the effectiveness of your model's performance. If your model delivers a positive result on validation data, go ahead with the current model. It rocks!

## A few common methods used for Cross Validation

There are various methods available for performing cross validation. I've discussed a few of them in this section.

### The validation set approach

In this approach, we reserve 50% of the dataset for validation and the remaining 50% for model training. However, a major disadvantage of this approach is that since we are training a model on only 50% of the dataset, there is a huge possibility that we might miss out on some interesting information about the data which will lead to a higher bias.

Python Code:

```
train, validation = train_test_split(data, test_size=0.50, random_state = 5)
```

R Code:

```
set.seed(101) # Set Seed so that same sample can be reproduced in future also

# Now Selecting 50% of data as sample from total 'n' rows of the data
sample <- sample.int(n = nrow(data), size = floor(.50*nrow(data)), replace = F)
train <- data[sample, ]
test  <- data[-sample, ]
```

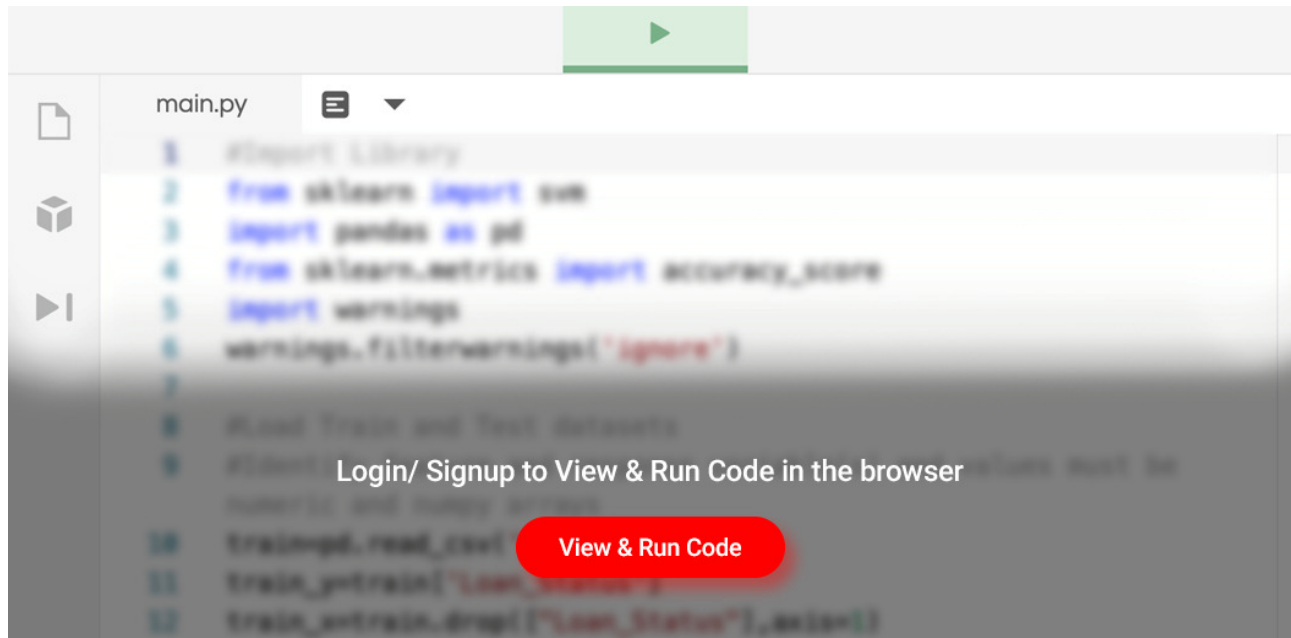
### Leave one out cross validation (LOOCV)

In this approach, we reserve only one data point from the available dataset, and train the model on the rest of the data. This process iterates for each data point. This also has its own advantages and disadvantages. Let's look at them:

- We make use of all data points, hence the bias will be low
- We repeat the cross validation process n times (where n is number of data points) which results in a higher execution time

- This approach leads to higher variation in testing model effectiveness because we test against one data point. So, our estimation gets highly influenced by the data point. If the data point turns out to be an outlier, it can lead to a higher variation

Python Code:



```
1 #Import Library
2 from sklearn import svm
3 import pandas as pd
4 from sklearn.metrics import accuracy_score
5 import warnings
6 warnings.filterwarnings("ignore")
7
8 #Load Train and Test datasets
9 #Load the data
10 train=pd.read_csv('train.csv')
11 train_y=train["Loan_Status"]
12 train_x=train.drop(["Loan_Status"],axis=1)
```

[https://id.analyticsvidhya.com/accounts/login/?next=https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/?&utm\\_source=coding-window-blog&source=coding-window-blog](https://id.analyticsvidhya.com/accounts/login/?next=https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/?&utm_source=coding-window-blog&source=coding-window-blog)

R Code:

```

score = list()

LOOCV_function = function(x,label){
  for(i in 1:nrow(x)){
    training = x[-i,]
    model = #... train model on training
    validation = x[i,]
    pred = predict(model, validation[,setdiff(names(validation),label)])
    score[[i]] = rmse(pred, validation[[label]]) # score/error of ith fold
  }
  return(unlist(score)) # returns a vector
}

```

LOOCV leaves one data point out. Similarly, you could leave  $p$  training examples out to have validation set of size  $p$  for each iteration. This is called LPOCV (Leave P Out Cross Validation)

## k-fold cross validation

From the above two validation methods, we've learnt:

1. We should train the model on a large portion of the dataset. Otherwise we'll fail to read and recognise the underlying trend in the data. This will eventually result in a higher bias
2. We also need a good ratio of testing data points. As we have seen above, less amount of data points can lead to a variance error while testing the effectiveness of the model
3. We should iterate on the training and testing process multiple times. We should change the train and test dataset distribution. This helps in validating the model effectiveness properly

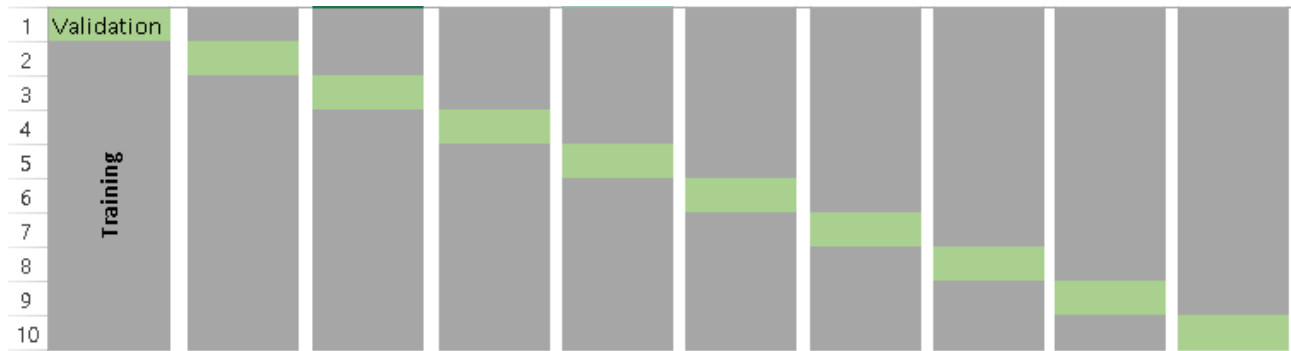
Do we have a method which takes care of all these 3 requirements?

Yes! That method is known as "**k-fold cross validation**". It's easy to follow and implement. Below are the steps for it:

1. Randomly split your entire dataset into  $k$  "folds"
2. For each  $k$ -fold in your dataset, build your model on  $k - 1$  folds of the dataset. Then, test the model to check the effectiveness for  $k$ th fold

3. Record the error you see on each of the predictions
4. Repeat this until each of the  $k$ -folds has served as the test set
5. The average of your  $k$  recorded errors is called the cross-validation error and will serve as your performance metric for the model

Below is the visualization of a  $k$ -fold validation when  $k=10$ .



(<https://www.analyticsvidhya.com/wp-content/uploads/2015/11/22.png>)

Now, one of most commonly asked questions is, “**How to choose the right value of  $k$ ?**”.

Always remember, a lower value of  $k$  is more biased, and hence undesirable. On the other hand, a higher value of  $K$  is less biased, but can suffer from large variability. It is important to know that a smaller value of  $k$  always takes us towards validation set approach, whereas a higher value of  $k$  leads to LOOCV approach.

Precisely, LOOCV is equivalent to  $n$ -fold cross validation where  $n$  is the number of training examples.

Python Code:

```
from sklearn.model_selection import KFold
kf = RepeatedKFold(n_splits=5, n_repeats=10, random_state=None)

for train_index, test_index in kf.split(X):
    print("Train:", train_index, "Validation:", test_index)
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
```



R code:

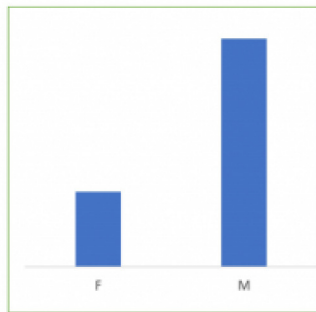
```
library(caret)
data(iris)

# Define train control for k fold cross validation
train_control <- trainControl(method="cv", number=10)
# Fit Naive Bayes Model
model <- train(Species~., data=iris, trControl=train_control, method="nb")
# Summarise Results
print(model)
```

#### **4. Stratified k-fold cross validation**

Stratification is the process of rearranging the data so as to ensure that each fold is a good representative of the whole. For example, in a binary classification problem where each class comprises of 50% of the data, it is best to arrange the data such that in every fold, each class comprises of about half the instances.

## Stratified K-Fold Cross Validation (K=5)



Class Distributions



It is generally a better approach when dealing with both bias and variance. A randomly selected fold might not adequately represent the minor class, particularly in cases where there is a huge class imbalance.

Python code snippet for stratified k-fold cross validation:

```
from sklearn.model_selection import StratifiedKFold
skf = StratifiedKFold(n_splits=5, random_state=None)
# X is the feature set and y is the target
for train_index, test_index in skf.split(X,y):
    print("Train:", train_index, "Validation:", val_index)
    X_train, X_test = X[train_index], X[val_index]
    y_train, y_test = y[train_index], y[val_index]
```

R Code:

```
library(caret)

# Folds are created on the basis of target variable

folds <- createFolds(factor(data$target), k = 10, list = FALSE)
```

Having said that, if the train set does not adequately represent the entire population, then using a stratified k-fold might not be the best idea. In such cases, one should use a simple **k-fold cross validation with repetition**.

In repeated cross-validation, the cross-validation procedure is repeated  $n$  times, yielding  $n$  random partitions of the original sample. The  $n$  results are again averaged (or otherwise combined) to produce a single estimation.

Python code for repeated k-fold cross validation:

```
from sklearn.model_selection import RepeatedKFold
rkf = RepeatedKFold(n_splits=5, n_repeats=10, random_state=None)
# X is the feature set and y is the target
for train_index, test_index in rkf.split(X):
    print("Train:", train_index, "Validation:", val_index)
    X_train, X_test = X[train_index], X[val_index]
    y_train, y_test = y[train_index], y[val_index]
```

## 5. Adversarial Validation

When dealing with real datasets, there are often cases where the test and train sets are very different. As a result, the internal cross-validation techniques might give scores that are not even in the ballpark of the test score. In such cases, adversarial validation offers an interesting solution.

The general idea is to check the degree of similarity between training and tests in terms of feature distribution. If it does not seem to be the case, we can suspect they are quite different. This intuition can be quantified by combining train and test sets, assigning 0/1 labels (0 – train, 1-test) and evaluating a binary classification task.

Let us understand, how this can be accomplished in the below steps:

1. Remove the target variable from the train set

```
train.drop(['target'], axis = 1, inplace = True)
```

2. Create a new target variable which is 1 for each row in the train set, and 0 for each row in the test set

```
train['is_train'] = 1  
test['is_train'] = 0
```

3. Combine the train and test datasets

```
df = pd.concat([train, test], axis = 0)
```

4. Using the above newly created target variable, fit a classification model and predict probabilities for each row to be in the test set

```
y = df['is_train']; df.drop('is_train', axis = 1, inplace = True)  
# Xgboost parameters  
xgb_params = {'learning_rate': 0.05,  
              'max_depth': 4,  
              'subsample': 0.9,  
              'colsample_bytree': 0.9,  
              'objective': 'binary:logistic',  
              'silent': 1,  
              'n_estimators': 100,  
              'gamma': 1,  
              'min_child_weight': 4}  
clf = xgb.XGBClassifier(**xgb_params, seed = 10)
```

5. Sort the train set using the calculated probabilities in step 4 and take top n% samples/rows as the validation set (n% is the fraction of the train set you want to keep in the validation set)

```
probs = clf.predict_proba(x1)[: ,1]
new_df = pd.DataFrame({'id':train.id, 'probs':probs})
new_df = new_df.sort_values(by = 'probs', ascending=False) # 30% validation set
val_set_ids = new_df.iloc[1:np.int(new_df.shape[0]*0.3),1]
```

*val\_set\_ids* will get you the ids from the train set that would constitute the validation set which is most similar to the test set. This will make your validation strategy more robust for cases where the train and test sets are highly dissimilar.

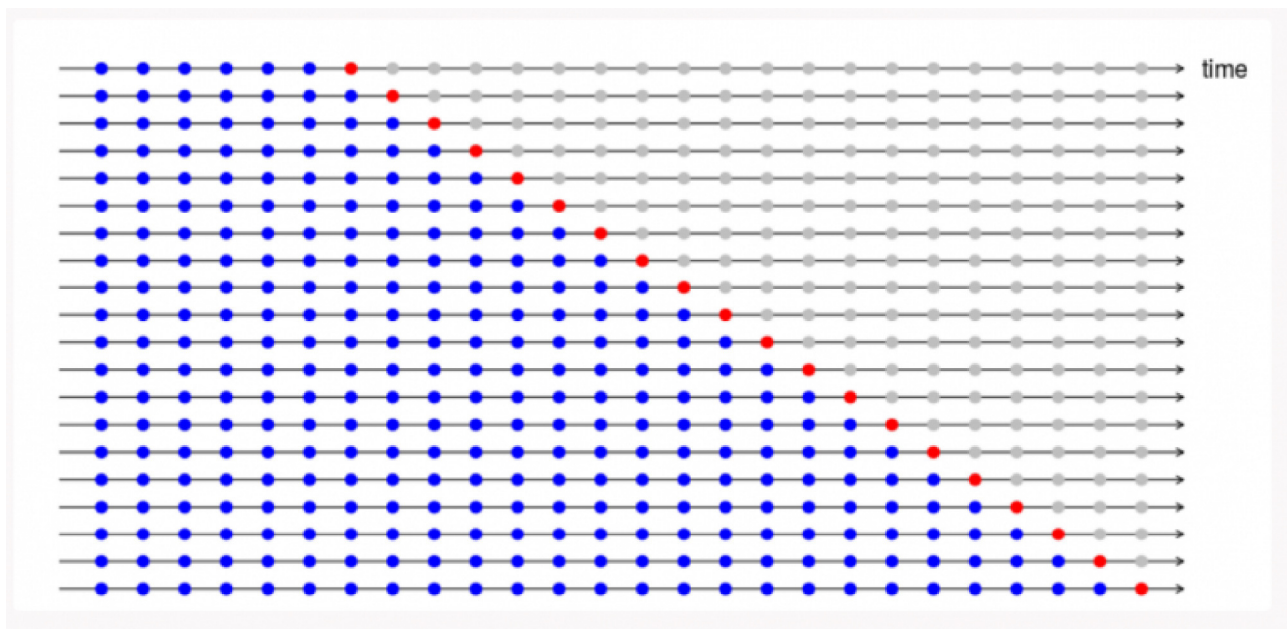
However, you must be careful while using this type of validation technique. Once the distribution of the test set changes, the validation set might no longer be a good subset to evaluate your model on.

## 6. Cross Validation for time series

Splitting a time-series dataset randomly does not work because the time section of your data will be messed up. For a time series forecasting problem, we perform cross validation in the following manner.

1. Folds for time series cross validation are created in a forward chaining fashion
2. Suppose we have a time series for yearly consumer demand for a product during a period of  $n$  years. The folds would be created like:

```
fold 1: training [1], test [2]
fold 2: training [1 2], test [3]
fold 3: training [1 2 3], test [4]
fold 4: training [1 2 3 4], test [5]
fold 5: training [1 2 3 4 5], test [6]
.
.
.
fold n: training [1 2 3 .... n-1], test [n]
```



We progressively select a new train and test set. We start with a train set which has a minimum number of observations needed for fitting the model. Progressively, we change our train and test sets with each fold. In most cases, 1 step forecasts might not be very important. In such instances, the forecast origin can be shifted to allow for multi-step errors to be used. For example, in a regression problem, the following code could be used for performing cross validation.

Python Code:

```
from sklearn.model_selection import TimeSeriesSplit
X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
y = np.array([1, 2, 3, 4])
tscv = TimeSeriesSplit(n_splits=3)
```

```
for train_index, test_index in tscv.split(X):
    print("Train:", train_index, "Validation:", val_index)
    X_train, X_test = X[train_index], X[val_index]
    y_train, y_test = y[train_index], y[val_index]
```

```
TRAIN: [0] TEST: [1]
```

```
TRAIN: [0 1] TEST: [2]
```

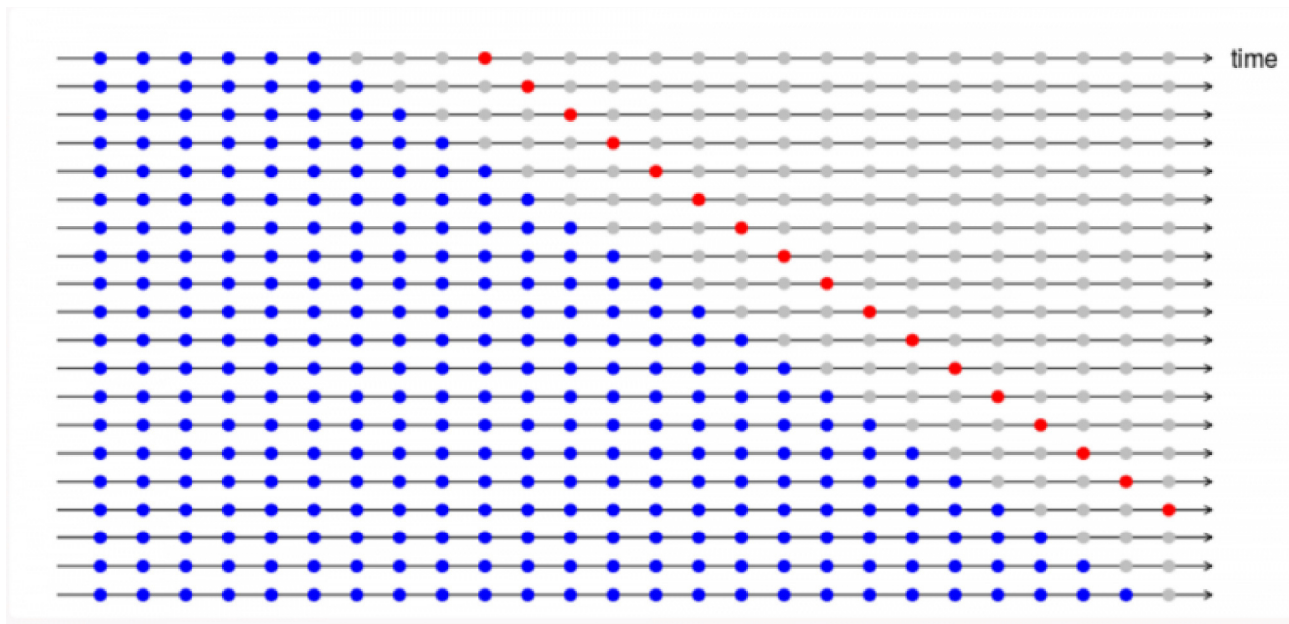
```
TRAIN: [0 1 2] TEST: [3]
```

R Code:

```
library(fpp)
library(forecast)
e <- tsCV(ts, Arima(x, order=c(2,0,0), h=1) #CV for arima model
sqrt(mean(e^2, na.rm=TRUE)) # RMSE
```

$h = 1$  implies that we are taking the error only for 1 step ahead forecasts.

( $h=4$ ) 4-step ahead error is depicted in the below diagram. This could be used if you want to evaluate your model for multi-step forecast.



## 7. Custom Cross Validation Techniques

Unfortunately, there is no single method that works best for all kinds of problem statements. Often, a custom cross validation technique based on a feature, or combination of features, could be created if that gives the user stable cross validation scores while making submissions in hackathons.

For example, in the recently finished contest '[Lord of the Machines](https://datahack.analyticsvidhya.com/contest/lord-of-the-machines/)' (<https://datahack.analyticsvidhya.com/contest/lord-of-the-machines/>) by Analytics Vidhya, the most stable validation technique used by the top finishers was using the campaign *id* variable.

### 3rd rank solution, by SRK and Mark:

Most of our time is spent on creating new features. We did validation split based on campaign ids. Our best single model is a light GBM that scored 0.7051 in LB. List of important features we used are:

1. Target encoding on the user ID, user ID - communication type
2. Min, max, mean and standard deviation of the mail sent time.
3. One hot encoding of the campaigns.
4. Time between current mail and previous mail
5. Number of campaigns inbetween current mail and previous mail
6. Total number of mail campaigns per user ID
7. Cumulative count of the mail at user level
8. Hour of the mail

Please have a look at the [problem statement](https://datahack.analyticsvidhya.com/contest/lord-of-the-machines/) (<https://datahack.analyticsvidhya.com/contest/lord-of-the-machines/>) and a few approaches discussed by the participants at this [thread](https://discuss.analyticsvidhya.com/t/share-your-approach-lord-of-machines/65061) (<https://discuss.analyticsvidhya.com/t/share-your-approach-lord-of-machines/65061>).

## How to measure the model's bias-variance?

After k-fold cross validation, we'll get  $k$  different model estimation errors ( $e_1, e_2, \dots, e_k$ ). In an ideal scenario, these error values should sum up to zero. To return the model's bias, we take the average of all the errors. Lower the average value, better the model.

Similarly for calculating the model variance, we take standard deviation of all the errors. A low value of standard deviation suggests our model does not vary a lot with different subsets of training data.

We should focus on achieving a balance between bias and variance. This can be done by reducing the variance and controlling bias to an extent. It'll result in a better predictive model. This trade-off usually leads to building less complex predictive models as well. For understanding bias-variance trade-off in more depth, please refer to section 9 of [this article](https://www.analyticsvidhya.com/blog/2017/06/a-comprehensive-guide-for-linear-ridge-and-lasso-regression/) (<https://www.analyticsvidhya.com/blog/2017/06/a-comprehensive-guide-for-linear-ridge-and-lasso-regression/>).



## End Notes

In this article, we discussed about overfitting and methods like cross-validation to avoid overfitting. We also looked at different cross-validation methods like validation set approach, LOOCV, k-fold cross validation, stratified k-fold and so on, followed by each approach's implementation in Python and R performed on the Iris dataset.


Did you find this article helpful? Please share your opinions/thoughts in the comments section below. And don't forget to test these techniques in AV's [hackathons](https://datahack.analyticsvidhya.com/contest/all/) (<https://datahack.analyticsvidhya.com/contest/all/>).


You can also read this article on Analytics Vidhya's Android APP





[https://play.google.com/store/apps/details?id=com.analyticsvidhya.android&utm\\_source=blog\\_article&utm\\_campaign=blog&pcampaignid=MKOther-global-all-co-prtnr-py-PartBadge-Mar2515-1](https://play.google.com/store/apps/details?id=com.analyticsvidhya.android&utm_source=blog_article&utm_campaign=blog&pcampaignid=MKOther-global-all-co-prtnr-py-PartBadge-Mar2515-1)


### Share this:

 (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/?share=linkedin&nb=1>)

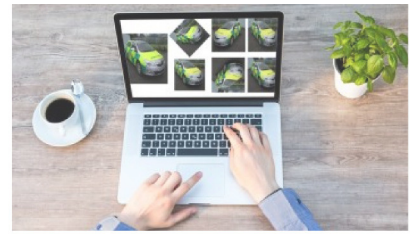
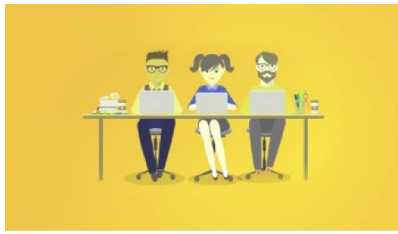
 (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/?share=facebook&nb=1>)

 (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/?share=twitter&nb=1>)

 (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/?share=pocket&nb=1>)

 (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/?share=reddit&nb=1>)

## Related Articles



(<https://www.analyticsvidhya.com/blog/2019/09/wins-analytics-wizard-2019-top-3-winners-solutions-from-our-biggest-data-science-hackathon/>) (<https://www.analyticsvidhya.com/blog/2019/10/5-winning-solutions-approaches-amexpert-retail-hackathon-feature-engineering/>) (<https://www.analyticsvidhya.com/blog/2019/12/image-augmentation-deep-learning-pytorch/>)

marketing-analytics-hackathon-top-3-inspiring-winning-solutions/)

WNS Analytics Wizard 2019:  
Top 3 Winners' Solutions from  
our Biggest Data Science  
Hackathon  
(<https://www.analyticsvidhya.com/blog/2019/09/wins-marketing-analytics-hackathon-top-3-inspiring-winning-solutions/>)

September 10, 2019  
In "Analytics Vidhya"

winning-solutions-approaches-amexpert-retail\_hackathon-feature-engineering/)

Top 5 Winning Solutions and  
Approaches from AmExpert  
2019 - Feature Engineering  
Special!  
([https://www.analyticsvidhya.com/blog/2019/10/5-winning-solutions-approaches-amexpert-retail\\_hackathon-feature-engineering/](https://www.analyticsvidhya.com/blog/2019/10/5-winning-solutions-approaches-amexpert-retail_hackathon-feature-engineering/))

October 21, 2019  
In "Analytics Vidhya"

augmentation-deep-learning-pytorch/)

Image Augmentation for Deep  
Learning using PyTorch -  
Feature Engineering for Images  
(<https://www.analyticsvidhya.com/blog/2019/12/image-augmentation-deep-learning-pytorch/>)

December 5, 2019  
In "Advanced"



TAGS : CROSS-VALIDATION ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/CROSS-VALIDATION/](https://www.analyticsvidhya.com/blog/tag/cross-validation/)), DATA HACK ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/DATA-HACK/](https://www.analyticsvidhya.com/blog/tag/data-hack/)), DATA SCIENCE COMPETITION ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/DATA-SCIENCE-COMPETITION/](https://www.analyticsvidhya.com/blog/tag/data-science-competition/)), KAGGLE ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/KAGGLE/](https://www.analyticsvidhya.com/blog/tag/kaggle/)), LIVE CODING ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/LIVE-CODING/](https://www.analyticsvidhya.com/blog/tag/live-coding/)), LOOCV ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/LOOCV/](https://www.analyticsvidhya.com/blog/tag/loocv/)), MACHINE LEARNING ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/MACHINE-LEARNING/](https://www.analyticsvidhya.com/blog/tag/machine-learning/)), OVERFITTING ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/OVERFITTING/](https://www.analyticsvidhya.com/blog/tag/overfitting/)), PREDICTIVE-MODEL ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/PREDICTIVE-MODEL/](https://www.analyticsvidhya.com/blog/tag/predictive-model/)), UNDERFITTING ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/UNDERFITTING/](https://www.analyticsvidhya.com/blog/tag/underfitting/)), VALIDATION ([HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/VALIDATION/](https://www.analyticsvidhya.com/blog/tag/validation/))

NEXT ARTICLE

## **Facebook Announces PyTorch 1.0 – A Major Release for Data Scientists and AI Researchers**

(<https://www.analyticsvidhya.com/blog/2018/05/facebook-announces-pytorch-1-0-open-source/>)

...

PREVIOUS ARTICLE

## **Top 5 GitHub Repositories and Reddit Discussions for Data Science & Machine Learning (April 2018)**

(<https://www.analyticsvidhya.com/blog/2018/05/top-5-github-reddit-data-science-machine-learning-april-2018/>)



(<https://www.analyticsvidhya.com/blog/author/sunil-ray/>)

**Sunil Ray**

**(<https://www.analyticsvidhya.com/blog/author/sunil-ray/>)**

I am a Business Analytics and Intelligence professional with deep experience in the Indian Insurance industry. I have worked for various multi-national Insurance companies in last 7 years.

This article is quite old and you might not get a prompt response from the author. We request you to post this comment on Analytics Vidhya's **Discussion portal** (<https://discuss.analyticsvidhya.com/>) to get your queries resolved

**29 COMMENTS**



**MUTHUPANDI**

[Reply](#)

November 19, 2015 at 6:43 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-99988>)

Thank you for good explained article.

Am just starting to explore the Analytics for past 6 months, Cross validation is the one i was looking to get my hands on and got ur article.

One question is what is the Error Function we need to use?



**AKASH (HTTP://--)**

[Reply](#)

November 19, 2015 at 6:01 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-100021>)

Thanks sunil for the article.

Is this the general way of writing the code in R and python for cross-validation? or are there other ways?

Thanks,



**NEEHAR**

[Reply](#)

November 20, 2015 at 1:42 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-100043>)

Hi Sunil ,

Thank you . Great article.



**NEEHAR**

[Reply](#)

November 20, 2015 at 1:49 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-100045>)

I have a question. if we are creating 10 fold cross validation, we are training a model on 10 different datasets.

It means we get 10 instances of model trained on 10 different datasets.

At the time of final prediction do we need to predict our data on these 10 instances of models ?

or take the skeleton of the model (same options we have used in CV) and train it on whole dataset and predict ?

Can anyone please clarify ?



**RAM**

[Reply](#)

November 22, 2015 at 4:57 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-100195>)

Neehar,

Good question.

Perhaps this will help you:

If you are working in R, you can use the caret library to do the same processing with fewer lines of code.

Note: The train function tries several models, and selects the best model.

```
### Load the library
```

```
library(caret)
```

```
# Load the iris dataset
```

```
data(iris)
```

```
# Define training control: 5 fold cross-validation. If you want to perform 10 fold cv, set  
number=10,
```

```
train_control <- trainControl(method="cv", number=5)
```

```
# Train the model using randomForest (rf)
```

```
model <- train(Sepal.Length~., data=iris, trControl=train_control, method="rf")
```

```
##The printed summary shows the sample sizes used, the best model selected and other  
information.
```

```
print(model)
```

```
# Make predictions
```

```
predictions <- predict(model, iris[,-1])
```

```
# Summarize results
result <- data.frame(Actual=iris[,1],Predicted=predictions)
result$Difference <- abs(result$Actual - result$Predicted)
summary(result$Difference)
```

## The End



**VENUGOPAL**

[Reply](#)

November 28, 2015 at 5:02 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-100676>)

Good one



**RAJESH ([HTTP://MYEXPERIMENTSWITHDATASCIENCE.BLOGSPOT.COM](http://myexperimentswithdatascience.blogspot.com))**

December 29, 2015 at 8:48 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-102731>)

Hi,

Nice article explaining k-fold cross validation. Just thought of adding the mae function from hydroGOF package

```
library(hydroGOF)
sim <- result$Predicted
obs <- result$Actual
mae(sim,obs)
```

Output : 0.3157678



**SELMI**

[Reply](#)

January 15, 2016 at 10:17 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-103810>)

thanks for your good article , i have a question if you can explain more please in fact : i have tested the two approaches of cross validation by using your script in the first hand and by using caret package as you mentioned in your comment : why in the caret package the sample sizes is always around 120,121...

it is possible to get for example sample sizes 140 or 90 ..

No pre-processing

Resampling: Cross-Validated (5 fold)

Summary of sample sizes: 120, 121, 119, 120, 120

Resampling results across tuning parameters:

any clarification please about functioning of this method.

thanks in advance



**RAM**

[Reply](#)

January 15, 2016 at 4:05 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-103829>)

@Semi,

: 'Why in the caret package the sample sizes is always around 120,121...'

A good question.

Answer: The 'sample size used by caret' depends on the resampling parameter provided in trainControl.

It seems that you used the value 5..

Try it again with a value 10. You will see a different sample size selected.

```
tc <- trainControl("cv", number=10)
```

```
model <- train(Sepal.Length~., data=iris, trControl=train_control, method="rf")
```

Hope this helps.



**SELM I**

[Reply](#)

January 15, 2016 at 4:49 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-103832>)

thanks for the reply but can you expaine to me "the resampling paramet"

when we use resamling is not a bootstrap ?

thanks

---



**RAM**

[Reply](#)

January 16, 2016 at 4:10 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-103914>)

The resampling method ('bootstrap' or 'no bootstrap') depends on the parameter specified in trainControl.

You can try the following, and see the results: For the benefit of others, please describe what you see.

```
tc.cv <- trainControl("cv", number=10)
model1 <- train(Sepal.Length~., data=iris, trControl=tc.cv, method="rf")
print (model1)
```

```
tc.boot <- trainControl("boot", number=10)
model2 <- train(Sepal.Length~., data=iris, trControl=tc.boot, method="rf")
print (model2)
```



**SELMi**

[Reply](#)

January 18, 2016 at 9:54 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-104012>)

@Ram thanks for your clarification , when i have compare two methos (cv and boot) , i remark that for model 1 (with cv,n=5 ) summary of sample sizes are: 120, 121, 120, 120, 119 and for the model 2 (with boot,n=5) Summary of sample sizes: 150, 150, 150, 150, 150

so can you telle me how can i use cv or boot method ? should i compare RMSE for each method and take the smaller value?

waiting your reply thanks in advance .



**RAM**

[Reply](#)

January 18, 2016 at 4:10 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-104026>)

@Selmi, You might want to read the article again. Sunil has explained it.  
(How to measure the model's bias-variance?)





**ASHISH YELKAR**

[Reply](#)

October 5, 2016 at 7:34 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-116809>)

Very explanation...keep it up...!!!

---



**TATSIANA**

[Reply](#)

April 14, 2017 at 7:47 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-126804>)

thank you for article! very helpful!

---



**SAMEER**

[Reply](#)

August 6, 2017 at 2:00 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-133745>)

Thanks Sunil, Very nice article..

R users can switch to using caret as library for cross validation. It's has a standardized wrapper for all the models and cross validation.. Traincontrl is the parameter name..

---



**PRIYANSHU**

[Reply](#)

March 8, 2018 at 12:14 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-151765>)

what if cross validation score becomes zero after the models are trained? Which conclusion we will come to? Model is appropriate or failed ??

---



**ANKIT CHOUDHARY**

[Reply](#)

June 5, 2018 at 5:41 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-153746>)

Hi Priyanshu,

What kind of problem are you trying to solve Regression/Classification?

What is the evaluation metric?

---



**ARPIT ([HTTPS://MACHINELEARNINGSTORIES.BLOGSPOT.IN](https://machinelearningstories.blogspot.in))**

[Reply](#)

March 29, 2018 at 8:42 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-152272>)

This is completely irrelevant topic. Model performance will never improve. Model Accuracy can be rightly captured by K fold validation. Final model is always on all data. So irrespective of using k fold, non k fold, final model will always be same.



**AISHWARYA SINGH**

[Reply](#)

April 4, 2018 at 1:42 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-152351>)

Hi Arpit,

It is correct that the final model will be on the whole dataset but using cross validation, we can tune our parameters effectively to improve the overall model performance.



**ANKIT CHOUDHARY**

[Reply](#)

April 4, 2018 at 7:37 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-152374>)

Great Article!



**MARIO**

[Reply](#)

May 14, 2018 at 8:34 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-153254>)

Great Article! The following part of the code placed on time series has a mistake:

```
from sklearn.model_selection import TimeSeriesSplit
```

```
from sklearn.model_selection import TimeSeriesSplit
```

```
X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
```

```
y = np.array([1, 2, 3, 4])
```

```
tscv = TimeSeriesSplit(n_splits=3)
```

```
for train_index, test_index in tscv.split(X):
    print("Train:", train_index, "Validation:", val_index)
    X_train, X_test = X[train_index], X[val_index]
    y_train, y_test = y[train_index], y[val_index]
```

-> val\_index is not defined it should be test\_index I think 😊

```
X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
y = np.array([1, 2, 3, 4])
tscv = TimeSeriesSplit(n_splits=3)
```

```
for train_index, test_index in tscv.split(X):
    print("Train:", train_index, "Validation:", test_index)
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
```



**FAIZAN SHAIKH**

[Reply](#)

May 17, 2018 at 3:56 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-153334>)

Thanks for the suggestion! We have updated the code.



**VIGNESH**

[Reply](#)

June 5, 2018 at 6:49 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-153740>)

Very informative Blog... Thanks Sunil!!



**SHANAL AGRAWAL**

[Reply](#)

June 6, 2018 at 10:59 am (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-153758>)

Thanks for a great article to understand the use of cross-validation techniques for a good fit of the model.



**DEB**

[Reply](#)

[July 7, 2018 at 10:46 am \(https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-154076\)](https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-154076)

Can anyone please answer what kind of split mechanism is ideal for a very imbalanced dataset in a classification problem ? Is it K-Fold, StratifiedKfold or ShuffleSplit ? I have the total dataset count of 3338. Out of which Class A count is 525, Class B counts 2134 and Class C counts 679. So, the dataset is having almost 70% of a particular class B and rest is divided amongst class A & C.

Please suggest. FYIP, I am using python.

Thanks



**PULKIT SHARMA**

[Reply](#)

[July 10, 2018 at 2:26 pm \(https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-154094\)](https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-154094)

Hi DEB,

You can use StratifiedkFold as it will keep equal ratio of each class in every split making each split similar.



**GEMMA**

[Reply](#)

[December 19, 2018 at 4:40 pm \(https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-156223\)](https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-156223)

Thanks @Sunil for this great article! Very useful 😊

About this code that @RAM wrote back in November 2015, my question is:

if I want to make sure my final model is not overfitted (even though RF seems to prevent overfitting), do I compare these results (code below) from the predictions, against the OOB error? I understand the former is the error in the CV trained datasets, and the latter is the error in the CV test datasets. Is this correct?

Many thanks!

```
# Make predictions
predictions <- predict(model, iris[,-1])
```

```
# Summarize results
result <- data.frame(Actual=iris[,1],Predicted=predictions)
result$Difference <- abs(result$Actual - result$Predicted)
summary(result$Difference)
```

---



**PULKIT SHARMA**

[Reply](#)

December 19, 2018 at 5:03 pm (<https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/#comment-156224>)

Hi Gemma,

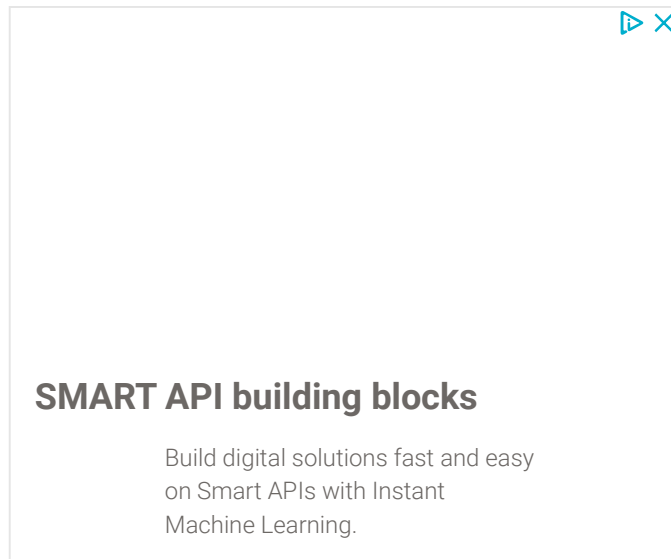
If the validation accuracy is in sync with the training accuracy, you can say that the model is not overfitting. If the training accuracy is increasing while the validation accuracy starts to decrease, then the model is overfitting.

---



## SMART API building blocks

Build digital solutions fast and easy  
on Smart APIs with Instant  
Machine Learning.



## RECOMMENDED READS

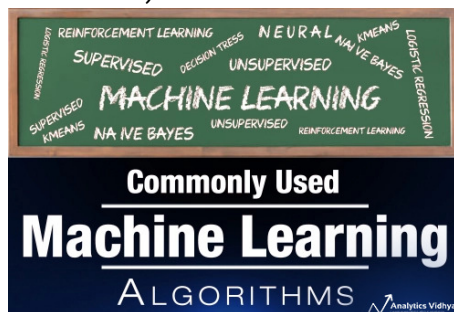
---



### A Complete Python Tutorial to Learn Data...

January 14, 2016

(<https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-learn-data-science-python-scratch-2/>)



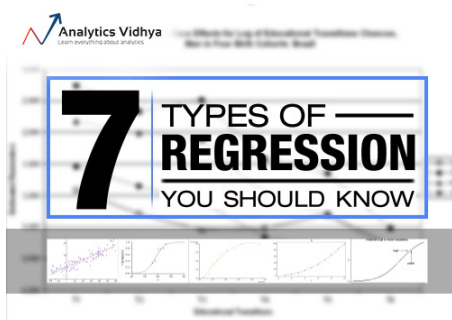
### Commonly used Machine Learning Algorithm...

September 9, 2017

(<https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/>)

### 7 Regression Techniques you should know!...

August 14, 2015



(<https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/>)

## RECOMMENDED RESOURCES

---



### Practice & Learn

Loan Prediction...

([https://datahack.analyticsvidhya.com/contest/practice-problem-loan-prediction-iii/?utm\\_source=Recommendation\\_resource&utm\\_medium=blog&utm\\_campaign=Novlist](https://datahack.analyticsvidhya.com/contest/practice-problem-loan-prediction-iii/?utm_source=Recommendation_resource&utm_medium=blog&utm_campaign=Novlist))



### Free Course for you

Python for Data Science...

([https://courses.analyticsvidhya.com/courses/introduction-to-data-science/?utm\\_source=Recommendation\\_resource&utm\\_medium=blog&utm\\_campaign=Novlist](https://courses.analyticsvidhya.com/courses/introduction-to-data-science/?utm_source=Recommendation_resource&utm_medium=blog&utm_campaign=Novlist))



### Try for free

Applied Machine Learning - Beginner to P...

([https://courses.analyticsvidhya.com/courses/applied-machine-learning-beginner-to-professional/?utm\\_source=Recommendation\\_resource&utm\\_medium=blog&utm\\_campaign=Novlist](https://courses.analyticsvidhya.com/courses/applied-machine-learning-beginner-to-professional/?utm_source=Recommendation_resource&utm_medium=blog&utm_campaign=Novlist))

## JOIN OUR COMMUNITY :

vidhya)

 $\gamma$