Building and Evaluating models

Building a statistical model

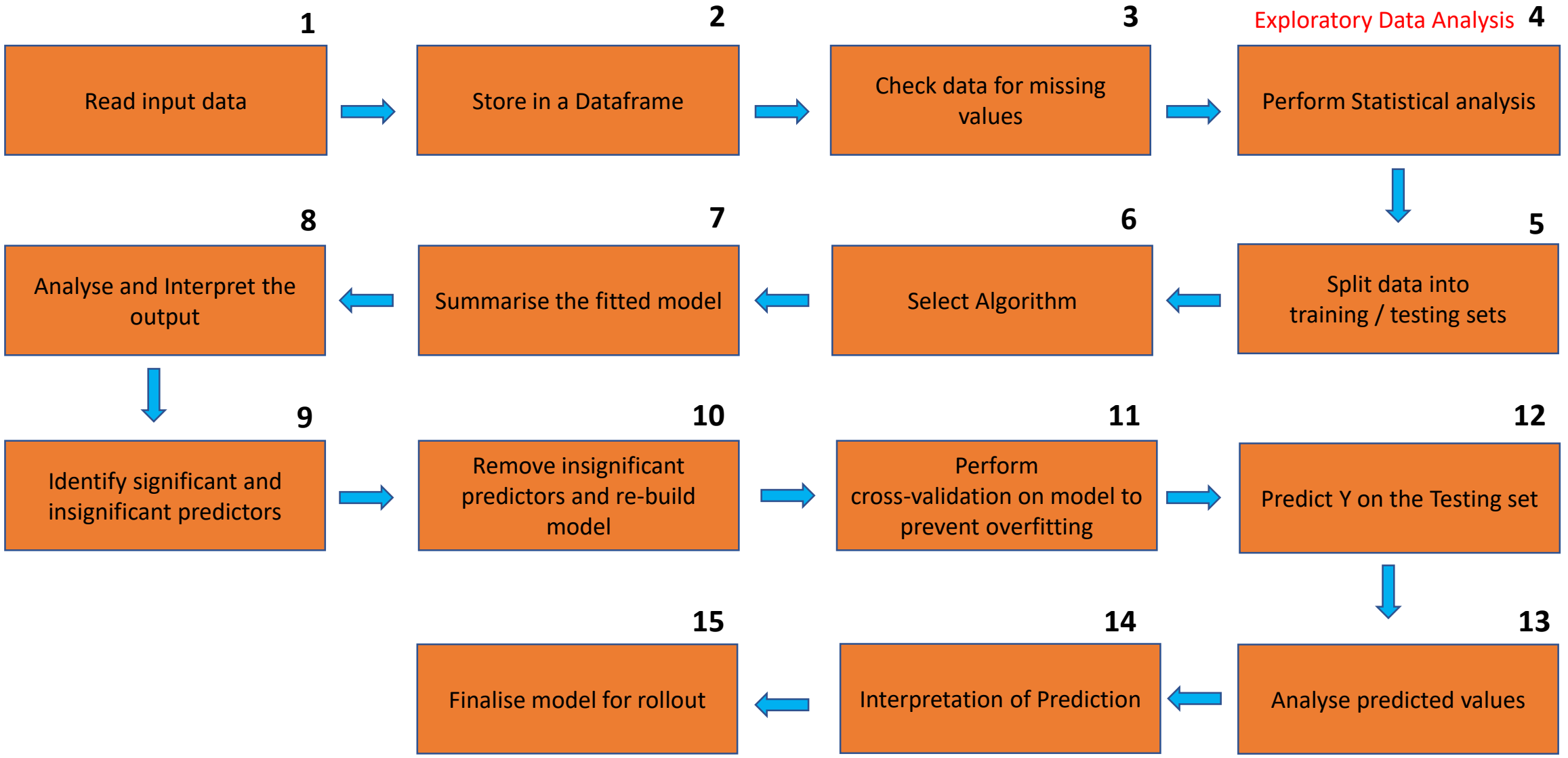
1) Model building process
2) Model summary
3) Cross validation
4) Underfitting and Overfitting
5) Parameter tuning
6) Confusion Matrix

1. Model building process

- Model building is a continuous process
- Multiple models should be built based on different factors like
  - ✓ Features (converting to appropriate data types, feature selection)
  - ✓ Parameter tuning
  - ✓ Standardization of parameters (depending upon the algorithm)
- Predictions are made on all the models and the best model is picked

- It is essential to have a good dataset before any model building process.
- High level of predictions can be achieved by having good data

Model building process



2. Model summary

- ✓ To check for errors during model building
- ✓ Identify significant and insignificant predictors (Linear and Logistic regression)
- ✓ Enables to view the split conditions (in Decision Trees), Support vectors (in SVM)

Linear regression

```
Call:
lm(formula = unpaid_tax ~ ., data = tax)

Residuals:
     Min       1Q   Median       3Q      Max
-0.29080 -0.11604 -0.09998  0.09102  0.44452

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -45.79635    4.87765  -9.389 8.29e-05 ***
lab_hrs       0.59697    0.08112   7.359 0.000323 ***
comp_hrs      1.17684    0.08407  13.998 8.29e-06 ***
reward        0.40511    0.04223   9.592 7.34e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2861 on 6 degrees of freedom
Multiple R-squared:  0.9834,    Adjusted R-squared:  0.9751
F-statistic: 118.5 on 3 and 6 DF,  p-value: 9.935e-06
```

Logistic regression

```
> cr_glm = glm(approved~creditscore, data=cr, family=binomial)
> summary(cr_glm)

Call:
glm(formula = approved ~ creditscore, family = binomial, data = cr)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.3821  -1.3353   0.9646   1.0019   1.3893

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.315679   1.768629  -0.744    0.457
creditscore  0.002539   0.002852   0.890    0.373

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 27.526  on 19  degrees of freedom
Residual deviance: 26.706  on 18  degrees of freedom
AIC: 30.706

Number of Fisher Scoring iterations: 4
```

Decision Tree

```
> ctree1

     Conditional inference tree with 16 terminal nodes

Response:  NSPF
Inputs:  LB, AC, FM
Number of observations:  1690

1) AC <= 0; criterion = 1, statistic = 239.069
  2) LB <= 136; criterion = 1, statistic = 126.202
    3) FM <= 14; criterion = 1, statistic = 29.757
      4) FM <= 9; criterion = 1, statistic = 25.09
        5) FM <= 1; criterion = 0.98, statistic = 9.985
          6) FM <= 0; criterion = 0.992, statistic = 11.88
            7)*  weights = 241
          6) FM > 0
            8)*  weights = 61
        5) FM > 1
          9)*  weights = 72
      4) FM > 9
        10)*  weights = 8
    3) FM > 14
      11)*  weights = 21
  2) LB > 136
    12) FM <= 1; criterion = 0.99, statistic = 11.435
      13) FM <= 0; criterion = 1, statistic = 20.66
        14)*  weights = 179
      13) FM > 0
        15)*  weights = 35
    12) FM > 1
      16) LB <= 146; criterion = 1, statistic = 22.253
        16) LB > 146
          18)*  weights = 17
1) AC > 0
  19) AC <= 1; criterion = 1, statistic = 45.306
    20) LB <= 136; criterion = 1, statistic = 26.057
      21) FM <= 7; criterion = 1, statistic = 37.599
        22) LB <= 129; criterion = 0.972, statistic = 9.35
          23)*  weights = 57
        22) LB > 129
          24)*  weights = 53
      21) FM > 7
        25)*  weights = 7
    20) LB > 136
      26)*  weights = 75
```

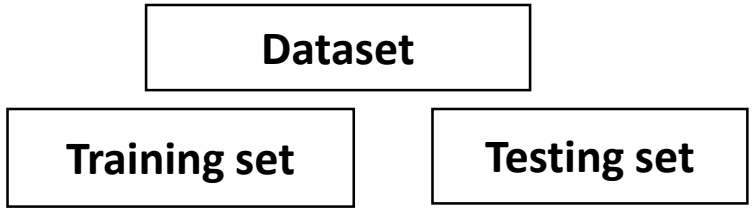
3. Cross-validation

- Model evaluation method
- An approximate estimate of how well the learned model will do on “unseen” data
- Slightly better than ‘residuals’
  - ➢ Residuals do not give a clear indication when predicting unseen data

Types of Cross validation

- Holdout method
- K-fold cross validation
- Leave-one-out cross validation

Holdout method

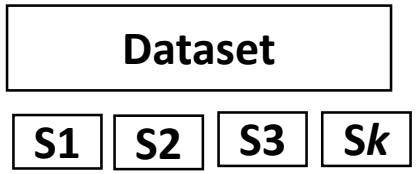


- Simple method
- Evaluation may have high variance *(depends on which set is in training and which set in testing)*

Leave one out cross validation

- N times the function is trained on all data except for one point and prediction is made for that one point
- Average error is calculated to evaluate

K-fold cross validation



*Holdout method **k** times*

Training set

One **k**-subset

Test set

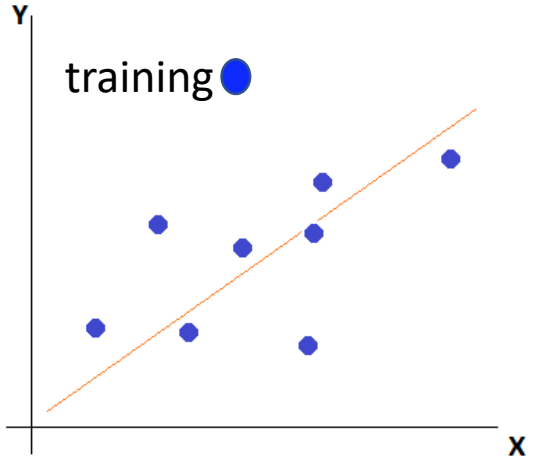
Other **k-1** subsets

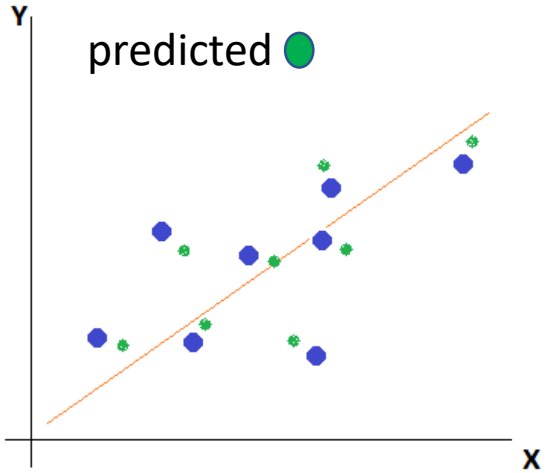
- Doesn't matter how data is divided
- Every data point gets to be in the training and test set
- Reduces variance
- On a large dataset, CV maybe time consuming

4. Underfitting and Overfitting

How well or how bad the predicted values fit the trained model

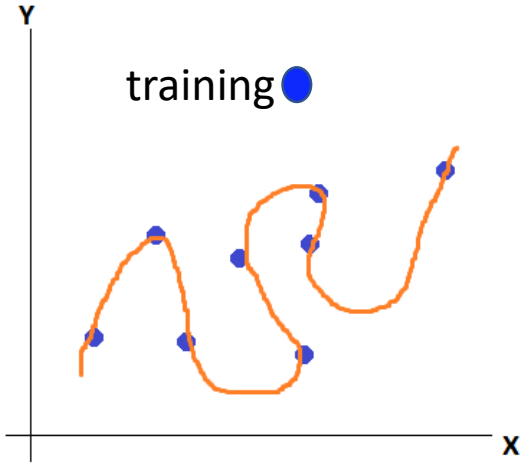
Simple model

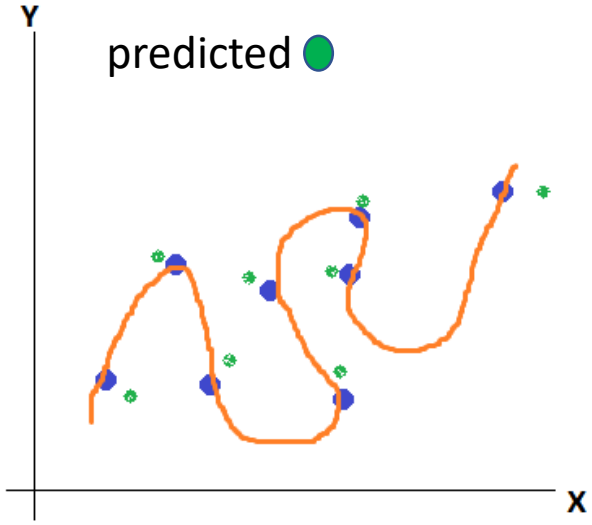




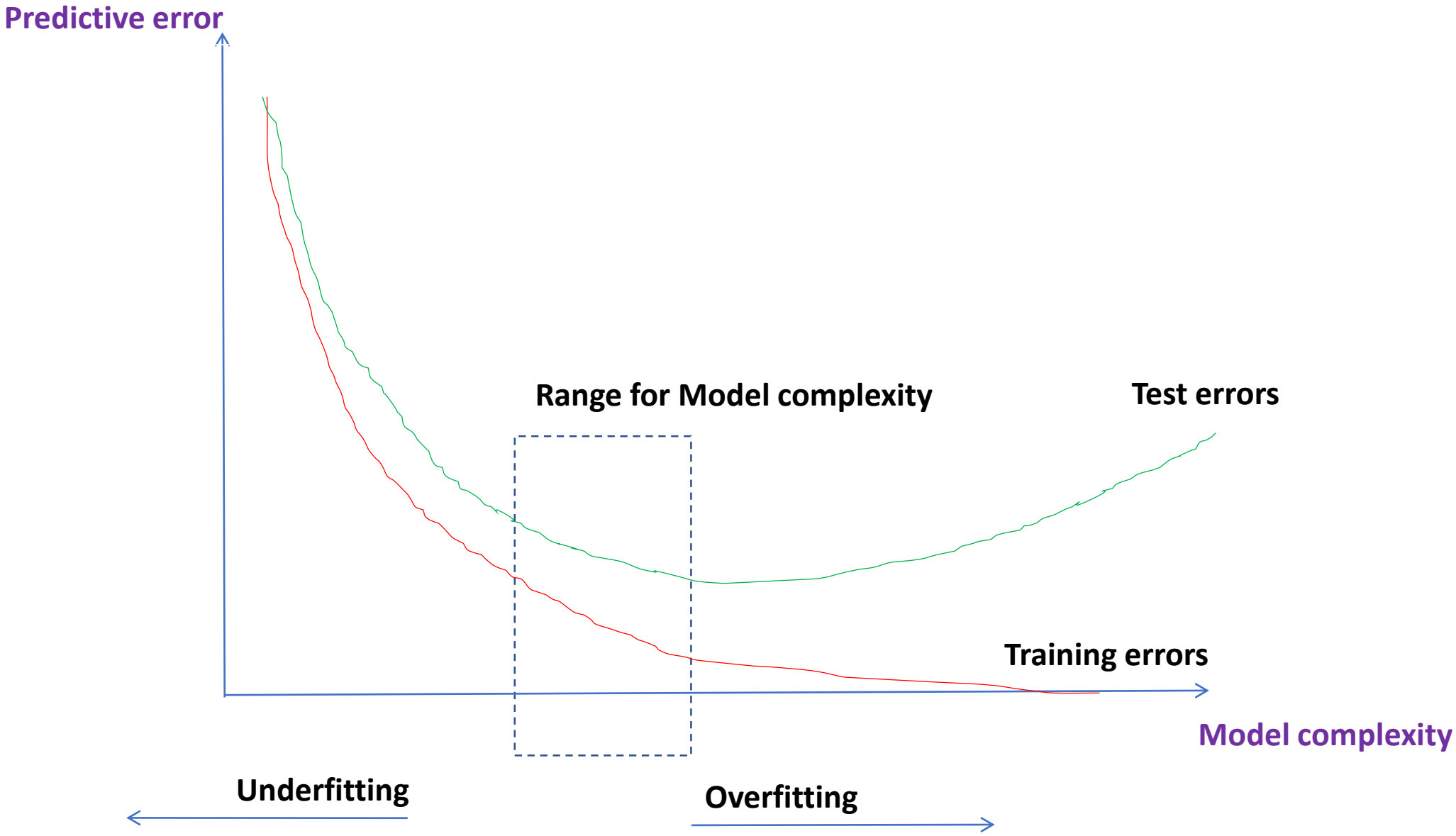
- Reasonably good prediction
- There are residuals, but variance isn't much
- Over-simplified models is **under-fitting**

Complex model





- Not a good Prediction model
- Expected and predicted values do not intersect
- This is **over-fitting**


Predictive error
Range for Model complexity
Test errors
Training errors
Model complexity
Underfitting
Overfitting

5. Parameters tuning

- Parameter tuning is a technique to tune the Hyperparameters of an algorithm for optimum performance
- Tuning depending on the dataset and algorithm used
- Some common examples
  - ✓ **Polynomial order:** Regression
  - ✓ **Number of nodes, pruning, trees** : Decision Trees and Random Forest
  - ✓ **Selection of Neighbours:** k-Nearest Neighbours
  - ✓ **Kernel type, Cost parameters**: Support Vector Machines

Model Evaluation techniques for Classification

A good model evaluation is done by predicting the model on multiple samples of test data

A model is considered GOOD when the results from each sample test data is consistent in the results (Accuracy etc..)

1. Confusion Matrix

2 classes

		Actual	
		Positive	**Negative**
Predicted	**Positive**	**TP**	**FP**
	Negative	**FN**	**TN**

TP: True positives: correct prediction
TN: True negatives: correct prediction
FN: False negatives: incorrect prediction
FP: False positives: incorrect prediction

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 732  20
         1 108  35

               Accuracy : 0.857
                 95% CI : (0.8323, 0.8793)
    No Information Rate : 0.9385
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2906
 Mcnemar's Test P-Value : 1.474e-14

            Sensitivity : 0.63636
            Specificity : 0.87143
         Pos Pred Value : 0.24476
         Neg Pred Value : 0.97340
             Prevalence : 0.06145
         Detection Rate : 0.03911
   Detection Prevalence : 0.15978
      Balanced Accuracy : 0.75390

       'Positive' Class : 1
```

It is important to select the right class as the '**positive**' class

Measures

Accuracy
(TP + TN)/(TP+TN+FP+FN)
Measure of all correct predictions

Error Rate (1 – Accuracy)
Measure of inaccurate predictions

Specificity
TN / (TN + FP)
It is a measure of actual negatives observations which are labelled (predicted) correctly
i.e. how many observations of negative class are labelled correctly. (when actually **no**, how many times is it correct?)

Precision
(Positive Predicted Value)
TP / (TP + FP)
It is a measure of **correctness** achieved in positive prediction

i.e. of observations labelled as positive, how many are actually labelled positive.

Recall (Sensitivity)
TP / (TP + FN)
(True Positive Rate TPR)
It is a measure of **actual positives** observations which are **labelled** (predicted) **correctly**.

i.e. how many observations of positive class are labelled correctly.

F measure
(2 × Precision x Recall) / (Precision + Recall)
It combines **Precision** and **Recall** as a measure of effectiveness of classification in terms of ratio of weighted importance on either recall or precision.
Higher the F-score, better the predictive power. $0 \leq F \leq 1$

For more details: https://en.wikipedia.org/wiki/Positive_and_negative_predictive_values

ROC Curve

Abbreviation for **Receiver Operating Characteristics**

Though, these methods are better than **accuracy** and **error** metrics, but still ineffective in answering the important questions on classification. For example:

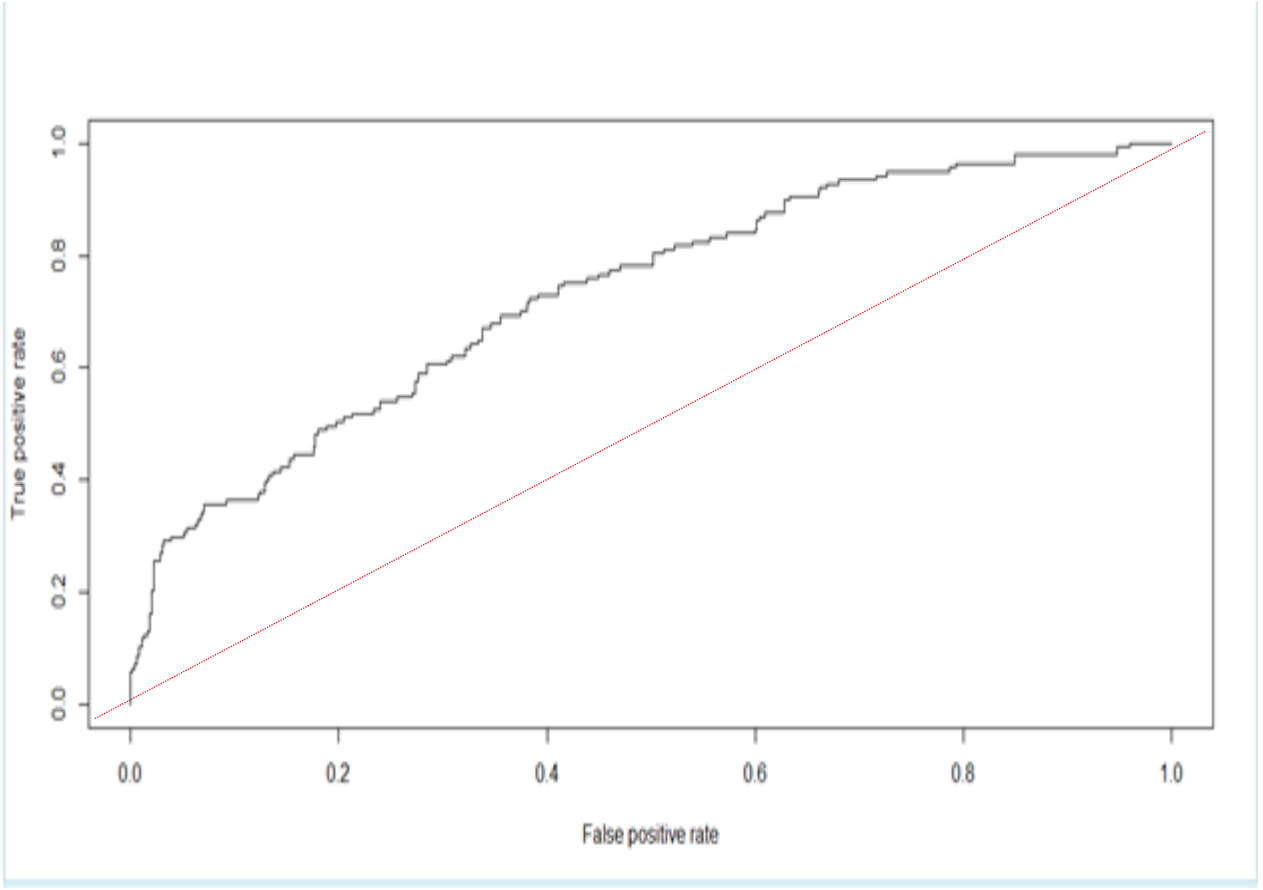
- **Precision does not tell us about negative prediction accuracy**
- **Recall is more interested in knowing actual positives**

This suggest, we can **still have a better metric to cater to our accuracy needs**

- ROC (Receiver Operating Characteristics) curve measures the accuracy of a classification prediction

ROC Curve / AUC (Area Under Curve)

- It's the most widely used evaluation metric for binary classification
- ROC Curve is formed by plotting **TP rate** (**Sensitivity (y-axis)**) and **FP rate** (**Specificity (x-axis**) for different cut-off values between 0 and 1. It is plotted against a random model (red dotted line)
- It is useful because it provides a visual representation of benefits (TP) and costs (FP) of a classification data
- The larger the area under ROC curve, higher will be the accuracy.
- Curve changes with the change in the cut-off value (eg: 0.1,0.2,0.3,0.4,0.5, etc.)
- ROC helps to determine the ideal cut-off value that will give the maximum accuracy



Kappa Statistic

- It is a measure that compares the **Observed Accuracy** and **Expected Accuracy**
- i.e. it measures how closely the ***predicted instances*** match with ***actual instances***

- **Formula**
  - kappa = (observed_accuracy – expected_accuracy) / (1-expected_accuracy)

- There is no standard way to interpret kappa value (values may vary according to context)

- May be a good measure to use for ***imbalanced classes***

- ***General guidelines. kappa value***
  - **> 0.75** – excellent
  - **0.4 – 0.75** – fair to good
  - **< 0.4** - poor

		Actual		
		Cats	Dogs	Total
Predicted	Cats	22	9	31
	Dogs	7	13	20
Total		29	22	**51**

- Observed_accuracy = (22+13)/51 = **0.69**
- Expected_accuracy = [(29*31)/51 + (22*20)/51] / 51 = **0.51**

- Kappa = (0.69 – 0.51) / (1 – 0.51) = **0.37**