

# Data Mining

# **Disciplina: Machine Learning**

**Prof. Carlos Eduardo Martins Relvas**

## **Coordenação:**

Prof. Dr. Adolpho Walter  
Pimazzi Canton

Profa. Dra. Alessandra de  
Ávila Montini

# Currículo

- Bacharel em Estatística, Universidade de São Paulo.
- Mestre em Estatística, Universidade de São Paulo.
- Doutorando em Ciência da Computação, Universidade de São Paulo.
- Itaú, 2010-2015. Principais atividades:
  - Consultoria estatística para várias áreas do banco com foco principal em melhorias no processo de modelagem de risco de crédito.
  - De 2013 a 2015, participação do projeto Big Data do banco usando tecnologia Hadoop e diversas técnicas de machine learning. Desenvolvemos diversos algoritmos em MapReduce usando R e Hadoop streaming, criando uma plataforma de modelagem estatística no Hadoop.
- Nubank, desde 2015. Equipe de Data Science, responsável por toda a parte de modelagem da empresa, desde modelos de crédito e até mesmo identificar motivos de atendimento.

# Agenda

- Bias and Variance
- Cross-Validation
- Pacote Caret
- Etapas de um projeto de machine learning

# Bias and Variance

# Bias and Variance

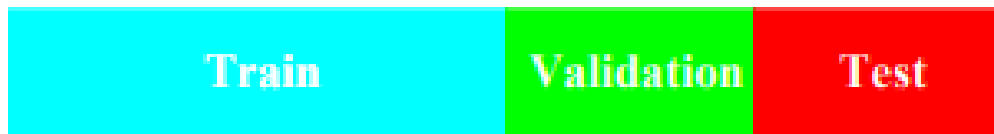
- Na maior parte dos algoritmos de Machine Learning, o objetivo é otimizar uma função de custo que reflete o erro do modelo utilizando os dados de treinamento. Geralmente usamos:

$$L(Y, \hat{f}(X)) = \begin{cases} (Y - \hat{f}(X))^2 & \text{squared error} \\ |Y - \hat{f}(X)| & \text{absolute error.} \end{cases}$$

- Porém o objetivo final de qualquer modelo é minimizar o erro em um novo conjunto de dados. Qual a melhor forma de estimarmos este erro?
- Infelizmente o erro na base de treino não é uma estimativa confiável.

# Bias and Variance

- Já vimos que quando estamos em uma situação com muitos dados, uma estratégia padrão é dividirmos os dados 3 (treino, validação e teste), em que:
- Treino: base de dados utilizada para treinar o modelo
- Validação: base de dados utilizada para escolher os hiperparâmetros dos algoritmos utilizados.
- Teste: base de dados utilizada para estimarmos o erro esperado em um novo conjunto de dados.



- Não há uma regra de como particionar estes dados, mas uma escolha típica é 50%, 25% e 25%.

# Bias and Variance

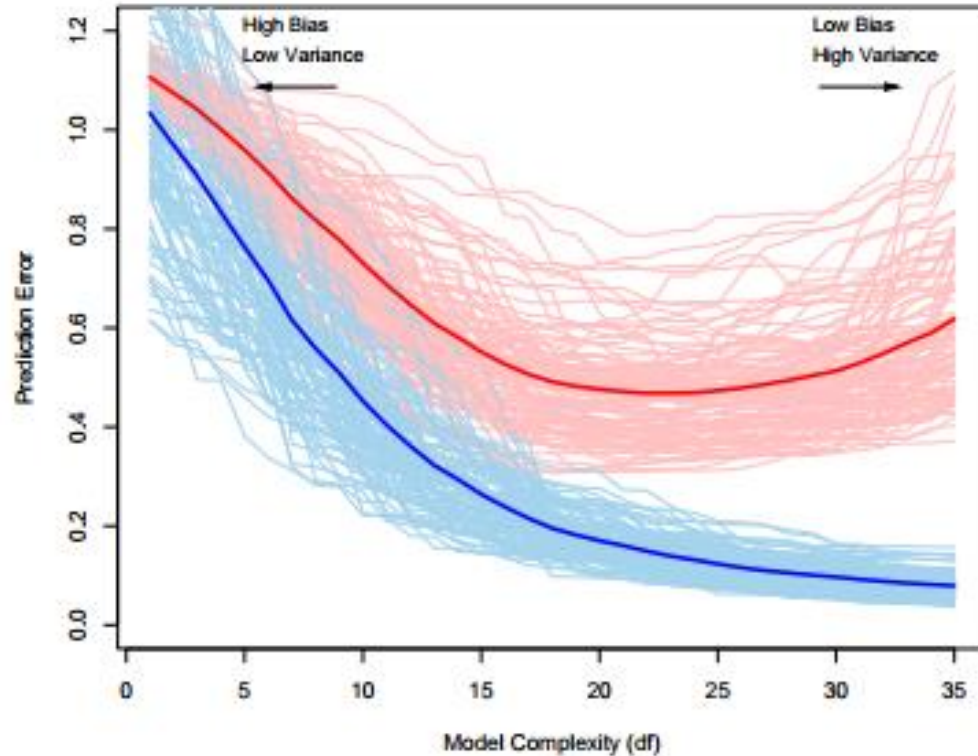
- Usando o erro quadrático, podemos escrever o erro esperado em uma nova base de dados como:

$$\begin{aligned}\text{Err}(x_0) &= E[(Y - \hat{f}(x_0))^2 | X = x_0] \\ &= \sigma_\epsilon^2 + [E\hat{f}(x_0) - f(x_0)]^2 + E[\hat{f}(x_0) - E\hat{f}(x_0)]^2 \\ &= \sigma_\epsilon^2 + \text{Bias}^2(\hat{f}(x_0)) + \text{Var}(\hat{f}(x_0)) \\ &= \text{Irreducible Error} + \text{Bias}^2 + \text{Variance}.\end{aligned}$$

- O primeiro termo se refere a aleatoriedade do gerador dos dados. Não importa o quanto nosso modelo seja bom, nunca capturaremos essa aleatoriedade.
- O bias reflete o quanto nossa estimativa está longe do valor verdadeiro.
- Por fim, temos a variância da nossa estimativa.
- Em geral, quanto mais complexo a solução proposta, reduzimos o bias, mas aumentamos a variância.

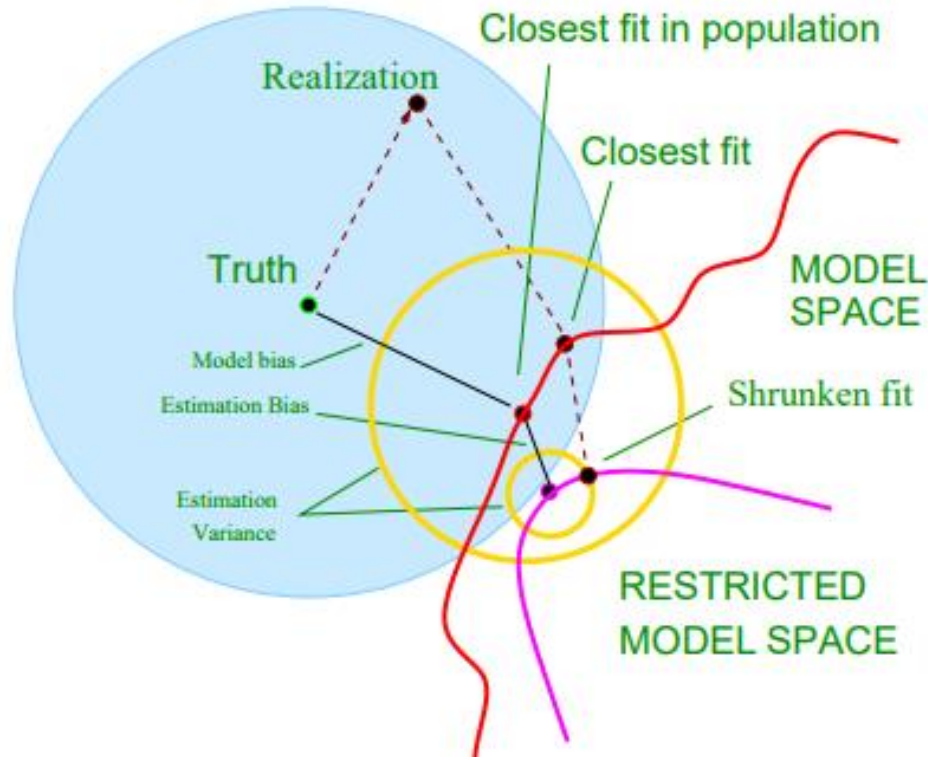


# Bias and Variance

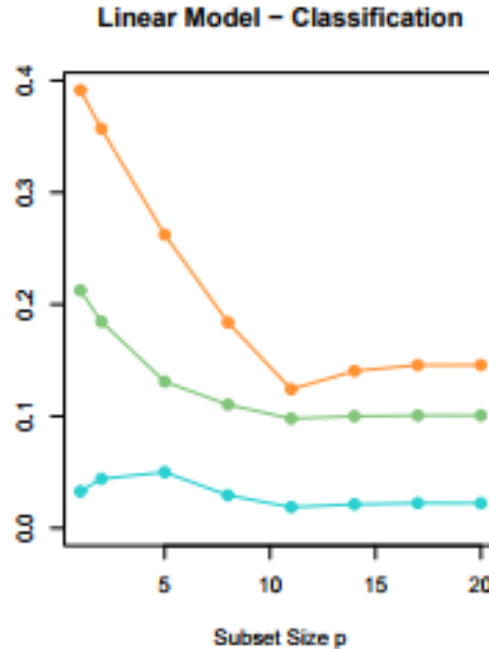
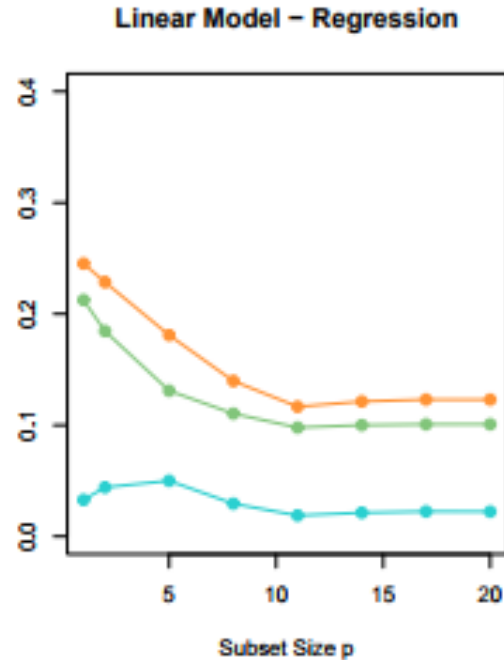


Linha vermelha refere-se ao erro na base de teste, enquanto que a linha azul reflete o erro na base de treino.

# Bias and Variance



# Bias and Variance



Laranja: Erro de predição  
Verde: Bias ao quadrado  
Azul: Variância

# Cross-validation

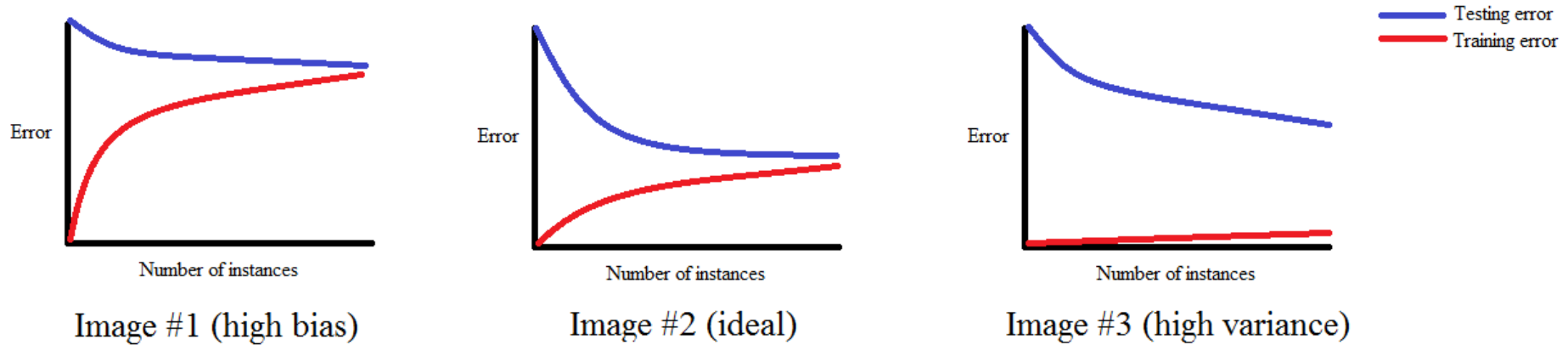
# Forma alternativa – Estimativa Erro

- Até agora durante o curso, sempre usamos a mesma estratégia de validação. Hoje vamos ver uma nova estratégia, conhecida como cross-validation.
- Cross-validation é um dos métodos mais simples e com certeza o mais utilizado para estimar o erro de predição.
- Este método estima diretamente o erro de generalização médio, ao aplicarmos nosso modelo preditivo em um novo conjunto de dados.

# K-fold Cross-validation

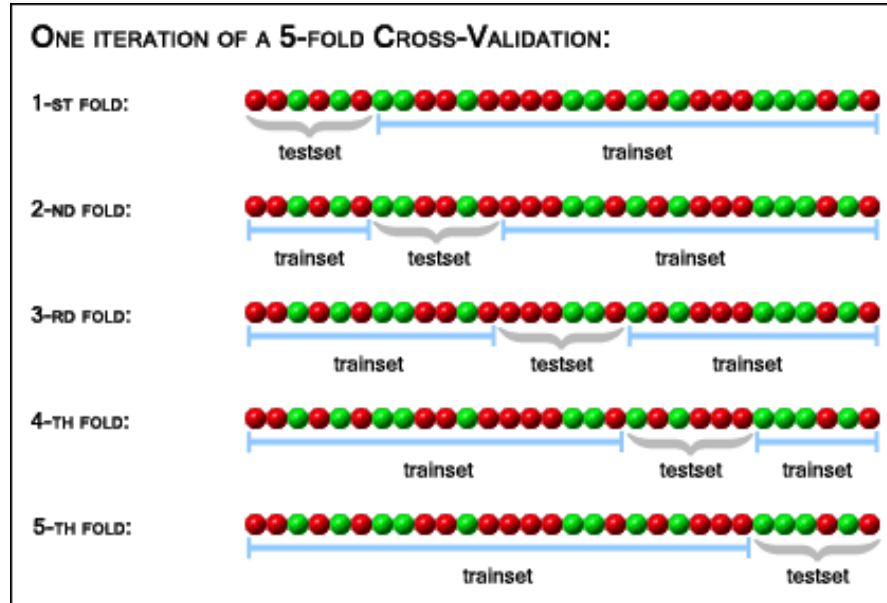
- A estratégia de cross validation é recomendada em duas situações:
  1. Quando não temos observações suficientes para particionar os dados em 3 pedaços (particionamos em apenas 2).
  2. Quando o modelo melhora com mais dados, isto é, se tivermos mais observações para treinar, o erro de generalização cai.

# Learning Curve



# K-fold Cross-validation

- Consiste em particionar os dados em K partes e realizar K modelos da seguinte forma:





# K-fold Cross-validation

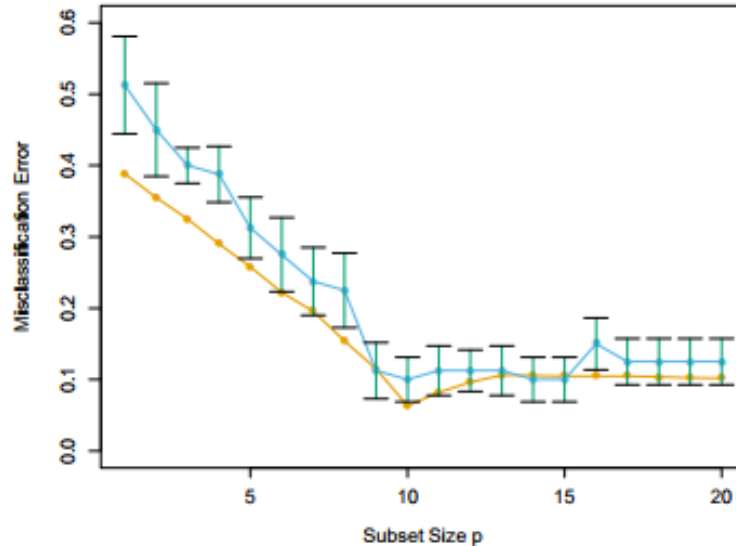
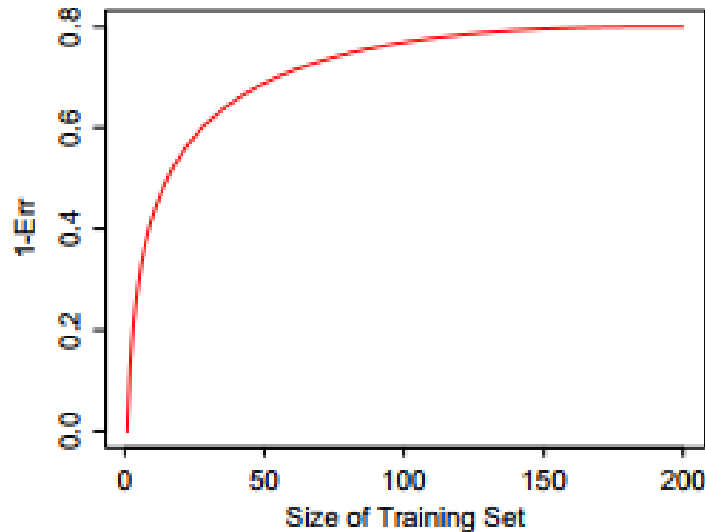
- Para o pedaço (fold)  $K$ , nós ajustamos o modelo nas outras  $K-1$  partes dos dados e calculamos o erro de previsão no fold  $K$ .
- Assim, teremos  $K$  diferentes performances. Agregando estes valores, temos uma boa estimativa do erro de predição em novos dados.
- Utilizamos cross-validation para a escolha de hiperparâmetros dos algoritmos, como o  $\lambda$  da regularização ou a profundidade das árvores.

# K-fold Cross-validation

- Qual valor escolhemos para K? Geralmente se use K entre 4 e 10. Mas quais os trade-offs?
- Com K igual a N (leave-one-out), o erro de CV é aproximadamente não viesado para o verdadeiro erro, mas pode ter alta variância devido a todos os ajustes serem muito similares, além de termos possíveis problemas computacionais em gerar N modelos.
- Com K igual a 5, CV tem uma variância menor, mas o viés pode ser um problema, dependendo de como o método varia com a quantidade de observações utilizadas.

# K-fold Cross-validation

- Se a learning curve não tiver convergido com o tamanho das observações de cada pedaço, o erro de CV pode ser super estimado.



# Cuidado!

- É muito comum as pessoas cometerem diversos erros ao realizar cross-validation. Por exemplo, se tivermos 50 amostras de classificação (25 bons e 25 maus) e 5000 variáveis. O que vocês acham da seguinte estratégia?
  1. Encontrar 100 variáveis mais correlacionadas com a resposta.
  2. Construir um classificador, como uma árvore.
  3. Realizar CV para encontrar os hiperparâmetros e ter uma estimativa do erro.

# Cuidado!

- É muito comum as pessoas cometerem diversos erros ao realizar cross-validation. Por exemplo, se tivermos 50 amostras de classificação (25 bons e 25 maus) e 5000 variáveis. O que vocês acham da seguinte estratégia?
1. Encontrar 100 variáveis mais correlacionadas com a resposta.
  2. Construir um classificador, como uma árvore.
  3. Realizar CV para encontrar os hiperparâmetros e ter uma estimativa do erro.

Errado!!

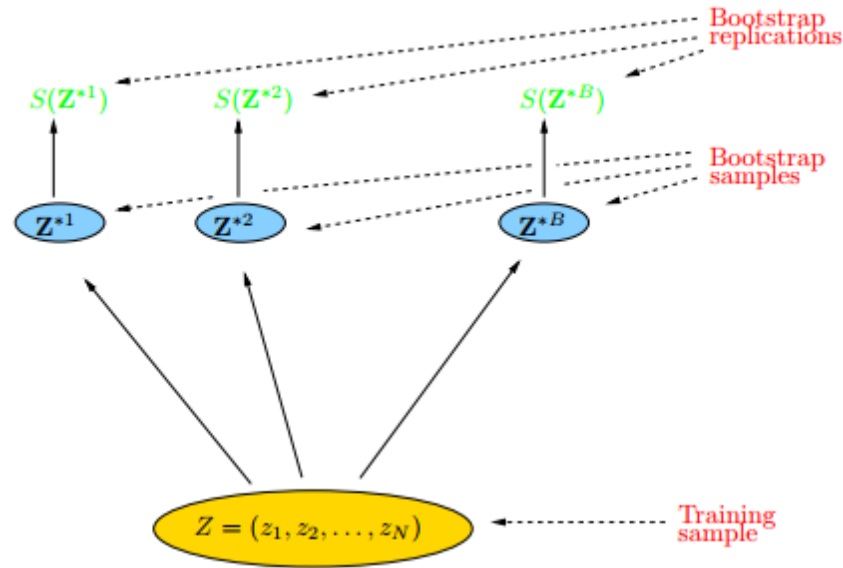
Como escolhemos as 100 variáveis usando todos os dados de treinamento, nossa estimativa de erro de CV estará sub-estimada.

# Cuidado!

- O modo correto seria pelo seguinte modo:
  1. Divida os dados de treinamento em K folds.
  2. Para cada fold  $k = 1, 2, \dots, K$ , faça:
    - a. Encontre as 100 melhores variáveis nos dados removendo o fold k.
    - b. Construa o modelo usando estas variáveis e usando os dados sem o fold k.
    - c. Calcule o erro utilizando apenas o fold k.
- Devemos fazer todas as etapas do modelo que possam dar uma vantagem indevida dentro do processo de CV.

# Forma alternativa – Estimativa Erro

- Outro método utilizado, embora um pouco menos utilizado, é o bootstrap.
- Já vimos como o bootstrap funciona no random forest.

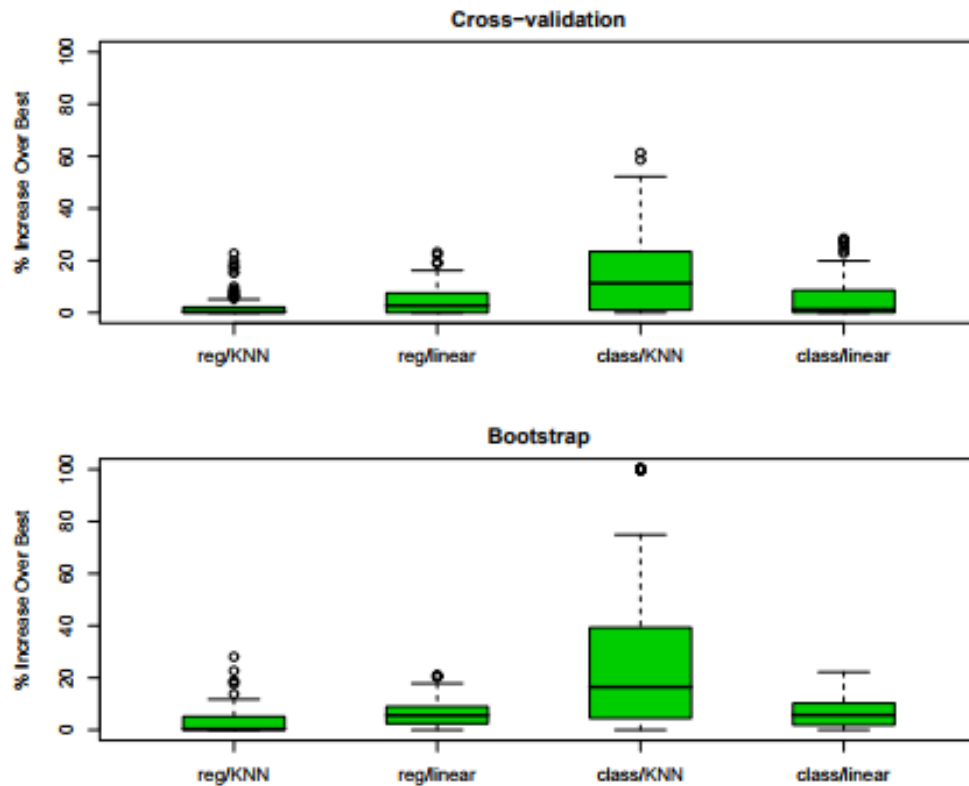


# Bootstrap

- Como podemos aplicar o bootstrap para termos uma estimativa do erro de generalização?
- Para cada base gerada pelo bootstrap, cerca de 63.2% dos dados originais estão presentes na base.
- Logo, podemos avaliar a performance de cada modelo gerado nos outros 36.8% como se fosse uma amostra de teste.
- Assim, a média de todas as amostras de “teste” representam uma boa estimativa para novos dados.



# Bootstrap x Cross-validation



# Laboratório

# Titanic

## Exemplos:

Estimar o erro de predição utilizando cross-validation.



Knowledge • 3,464 teams

## Titanic: Machine Learning from Disaster

Fri 28 Sep 2012

Sat 31 Dec

Dashboard

Home



Data



Make a submission



Competition Details » [Get the Data](#) » [Make a submission](#)

## Predict survival on the Titanic using

# Titanic

## VARIABLE DESCRIPTIONS:

survival	Survival (0 = No; 1 = Yes)
pclass	Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
name	Name
sex	Sex
age	Age
sibsp	Number of Siblings/Spouses Aboard
parch	Number of Parents/Children Aboard
ticket	Ticket Number
fare	Passenger Fare
cabin	Cabin
embarked	Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

**Exercícios:** <http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29>



## Breast Cancer Wisconsin (Diagnostic) Data Set

Download: [Data Folder](#) [Data Set Description](#)

Abstract: Diagnostic Wisconsin Breast Cancer Database



Data Set Characteristics:	Multivariate	Number of Instances:	569	Area:	Life
Attribute Characteristics:	Real	Number of Attributes:	32	Date Donated	1995-11-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	390512

# CV

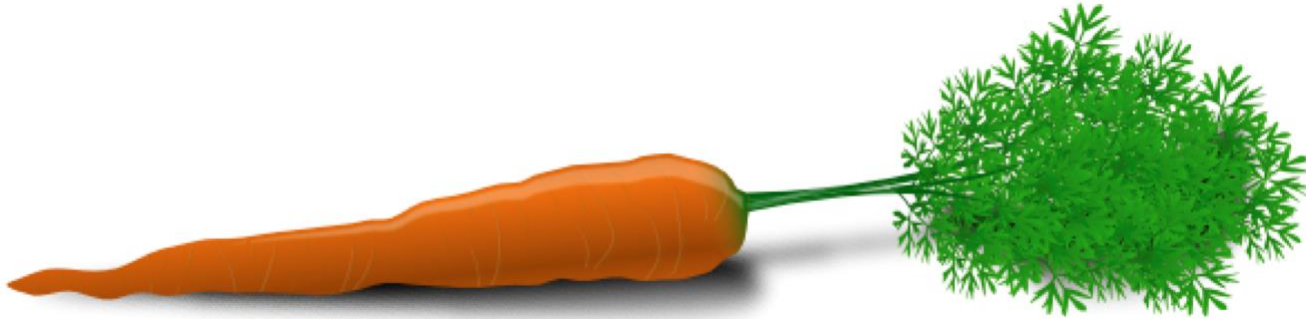
- 1.) Utilize seed de 42 e crie amostras de treino (70%) e teste (30%).
- 2.) Estime o erro de generalização usando cross-validation.
- 3.) Interprete os resultados.

# Pacote Caret

# Caret

O pacote caret (classification and regression training) é um conjunto de funções que tenta padronizar o processo de criação de modelos preditivos.

No R temos diversos pacotes para modelos a criação de modelos supervisionados. Infelizmente, não há uma notação única e há algumas variações de como chamar cada uma das funções. O caret padroniza isso, além de criar várias novas funções úteis para o processo de modelagem.

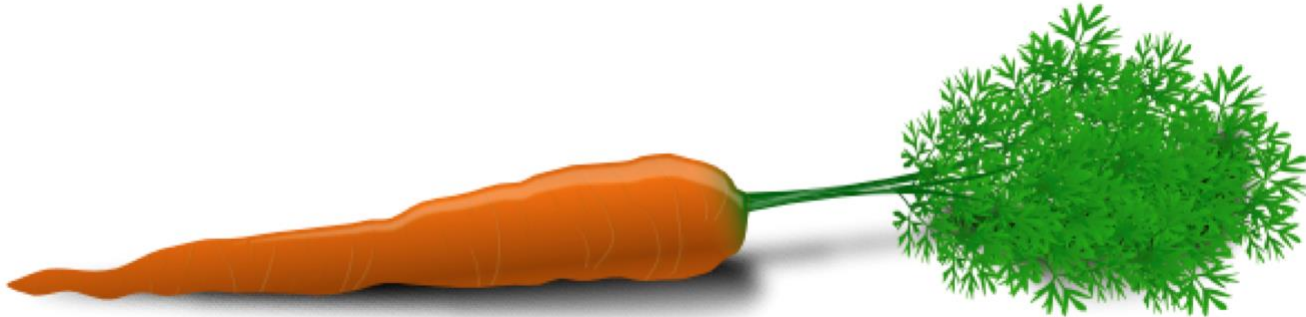




# Caret

O pacote caret apresenta funções para:

- Partição dos dados (treino, validação, etc);
- Pré-processamento (imputação de missings, transformações, etc);
- Seleção de variáveis.
- Model tuning.
- Variable importance.



# Caret

Model	method Argument Value	Type	Packages	Tuning Parameters
Boosted Classification Trees	ada	Classification	ada, plyr	iter, maxdepth, nu
Bagged AdaBoost	AdaBag	Classification	adabag, plyr	mfinal, maxdepth
AdaBoost.M1	AdaBoost.M1	Classification	adabag, plyr	mfinal, maxdepth, coeflearn
Adaptive Mixture Discriminant Analysis	amdai	Classification	adaptDA	model
Adaptive-Network-Based Fuzzy Inference System	ANFIS	Regression	frbs	num.labels, max.iter
Model Averaged Neural Network	avNNet	Dual Use	nnet	size, decay, bag
Naive Bayes Classifier with Attribute Weighting	awnb	Classification	bnclassify	smooth
Tree Augmented Naive Bayes Classifier with Attribute Weighting	awtan	Classification	bnclassify	score, smooth
Bagged Model	bag	Dual Use	caret	vars
Bagged MARS	bagEarth	Dual Use	earth	nprune, degree
Bagged MARS using gCV Pruning	bagEarthGCV	Dual Use	earth	degree
Bagged Flexible Discriminant Analysis	bagFDA	Classification	earth, mda	degree, nprune
Bagged FDA using gCV Pruning	bagFDAGCV	Classification	earth	degree
Bayesian Additive Regression Trees	bartMachine	Dual Use	bartMachine	num_trees, k, alpha, beta, nu
Bayesian Generalized Linear Model	bayesglm	Dual Use	arm	None
Self-Organizing Map	bdk	Dual Use	kohonen	xdim, ydim, xweight, topo
Binary Discriminant Analysis	binda	Classification	binda	lambda.freqs
Boosted Tree	blackboost	Dual Use	party, mboost, plyr	mstop, maxdepth
The Bayesian lasso	blasso	Regression	monomvn	sparsity
Bayesian Ridge Regression (Model Averaged)	blassoAveraged	Regression	monomvn	None
Random Forest with Additional Feature Selection	Boruta	Dual Use	Boruta, randomForest	mtry

# Caret

Bayesian Ridge Regression	bridge	Regression	monomvn	None
Bayesian Regularized Neural Networks	brnn	Regression	brnn	neurons
Boosted Linear Model	BstLm	Dual Use	bst, plyr	mstop, nu
Boosted Smoothing Spline	bstSm	Dual Use	bst, plyr	mstop, nu
Boosted Tree	bstTree	Dual Use	bst, plyr	mstop, maxdepth, nu
C5.0	C5.0	Classification	C50, plyr	trials, model, winnow
Cost-Sensitive C5.0	C5.0Cost	Classification	C50, plyr	trials, model, winnow, cost
Single C5.0 Ruleset	C5.0Rules	Classification	C50	None
Single C5.0 Tree	C5.0Tree	Classification	C50	None
Conditional Inference Random Forest	cforest	Dual Use	party	mtry
CHI-squared Automated Interaction Detection	chaid	Classification	CHAID	alpha2, alpha3, alpha4
SIMCA	CSimca	Classification	rrcovHD	None
Conditional Inference Tree	ctree	Dual Use	party	mincriterion
Conditional Inference Tree	ctree2	Dual Use	party	maxdepth
Cubist	cubist	Regression	Cubist	committees, neighbors
Dynamic Evolving Neural-Fuzzy Inference System	DENFIS	Regression	frbs	Dthr, max.iter
Stacked AutoEncoder Deep Neural Network	dnn	Dual Use	deepnet	layer1, layer2, layer3, hidden_dropout, visible_dropout
Linear Distance Weighted Discrimination	dwdLinear	Classification	kerndwd	lambda, qval
Distance Weighted Discrimination with Polynomial Kernel	dwdPoly	Classification	kerndwd	lambda, qval, degree, scale
Distance Weighted Discrimination with Radial Basis Function Kernel	dwdRadial	Classification	kernlab, kerndwd	lambda, qval, sigma

# Caret

## Exemplos:

Prever a probabilidade de sobrevivência dos passageiros do Titanic.



Knowledge • 3,464 teams

## Titanic: Machine Learning from Disaster

Fri 28 Sep 2012

Sat 31 Dec 2012

Dashboard

Home



Data



Make a submission



Competition Details » [Get the Data](#) » [Make a submission](#)

## Predict survival on the Titanic using

## Exemplos:

# Caret

Prever a probabilidade de sobrevivência dos passageiros do Titanic.

### VARIABLE DESCRIPTIONS:

survival	Survival (0 = No; 1 = Yes)
pclass	Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
name	Name
sex	Sex
age	Age
sibsp	Number of Siblings/Spouses Aboard
parch	Number of Parents/Children Aboard
ticket	Ticket Number
fare	Passenger Fare
cabin	Cabin
embarked	Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

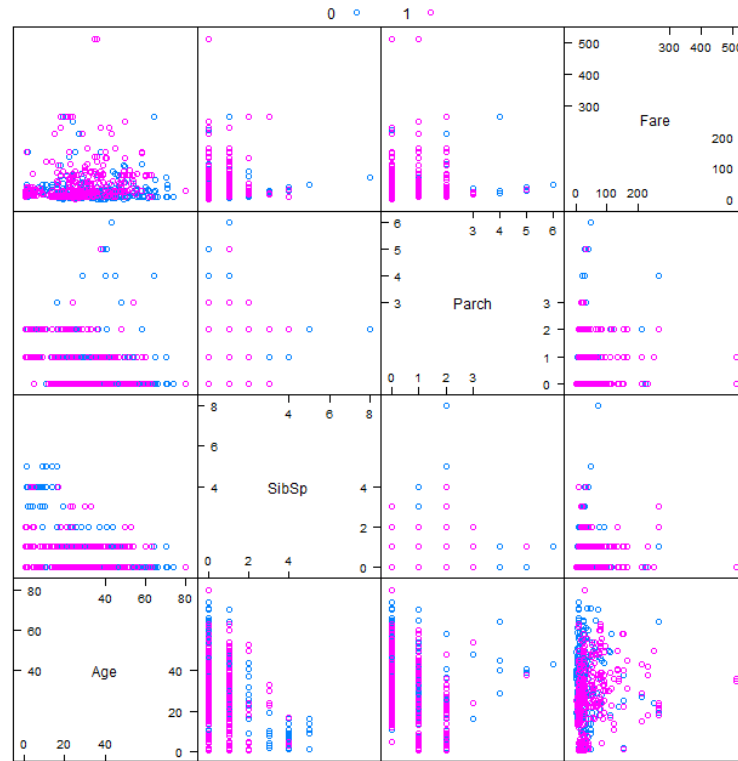
# Caret

```
library(caret)
featurePlot(x = dados[, c("Age", "SibSp", "Parch", "Fare")],
            y = as.factor(dados$Survived),
            plot = "pairs", auto.key = list(columns = 2))

featurePlot(x = dados[, c("Age", "SibSp", "Parch", "Fare")],
            y = as.factor(dados$Survived), layout = c(4,1 ),
            plot = "box", auto.key = list(columns = 2),
            scales = list(y = list(relation="free"),
                           x = list(rot = 90)))

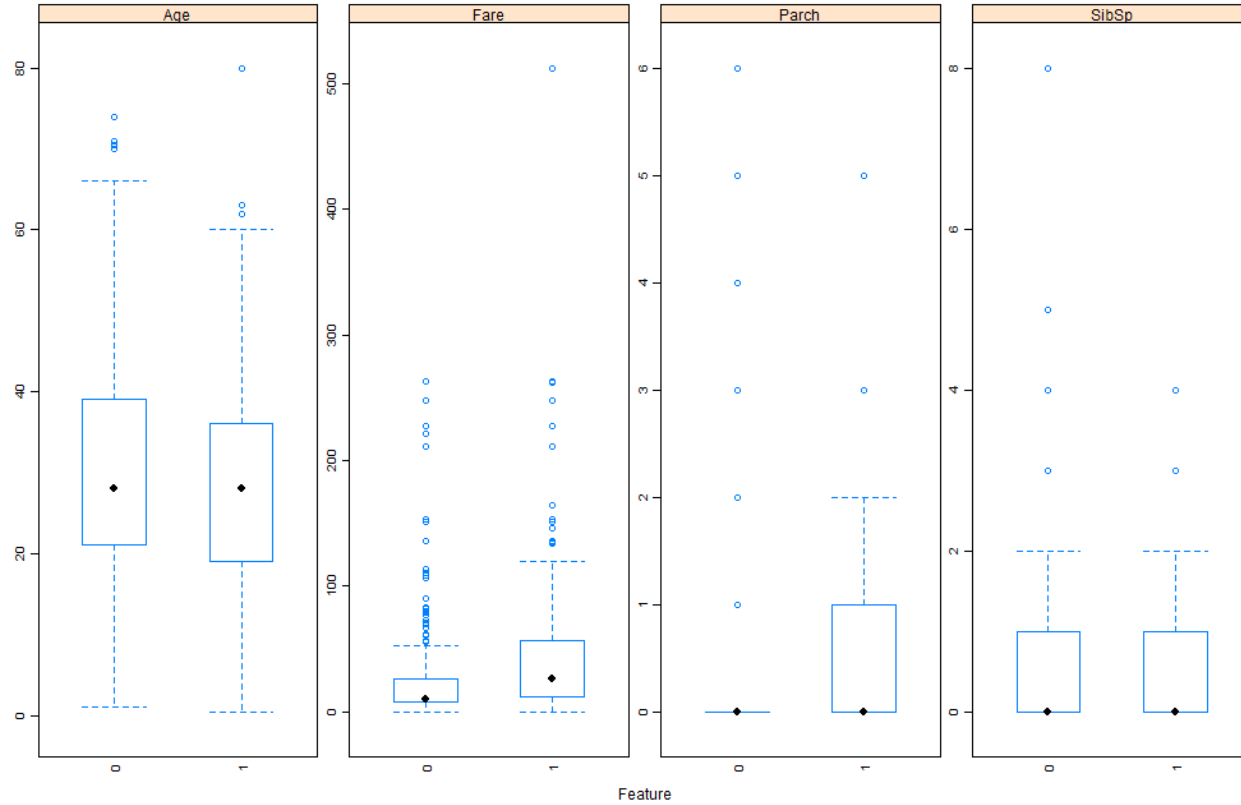
featurePlot(x = dados[, c("Age", "SibSp", "Parch", "Fare")],
            y = as.factor(dados$Survived), layout = c(4,1 ),
            plot = "density", auto.key = list(columns = 2),
            scales = list(y = list(relation="free"),
                           x = list(relation = "free")))
```

# Caret



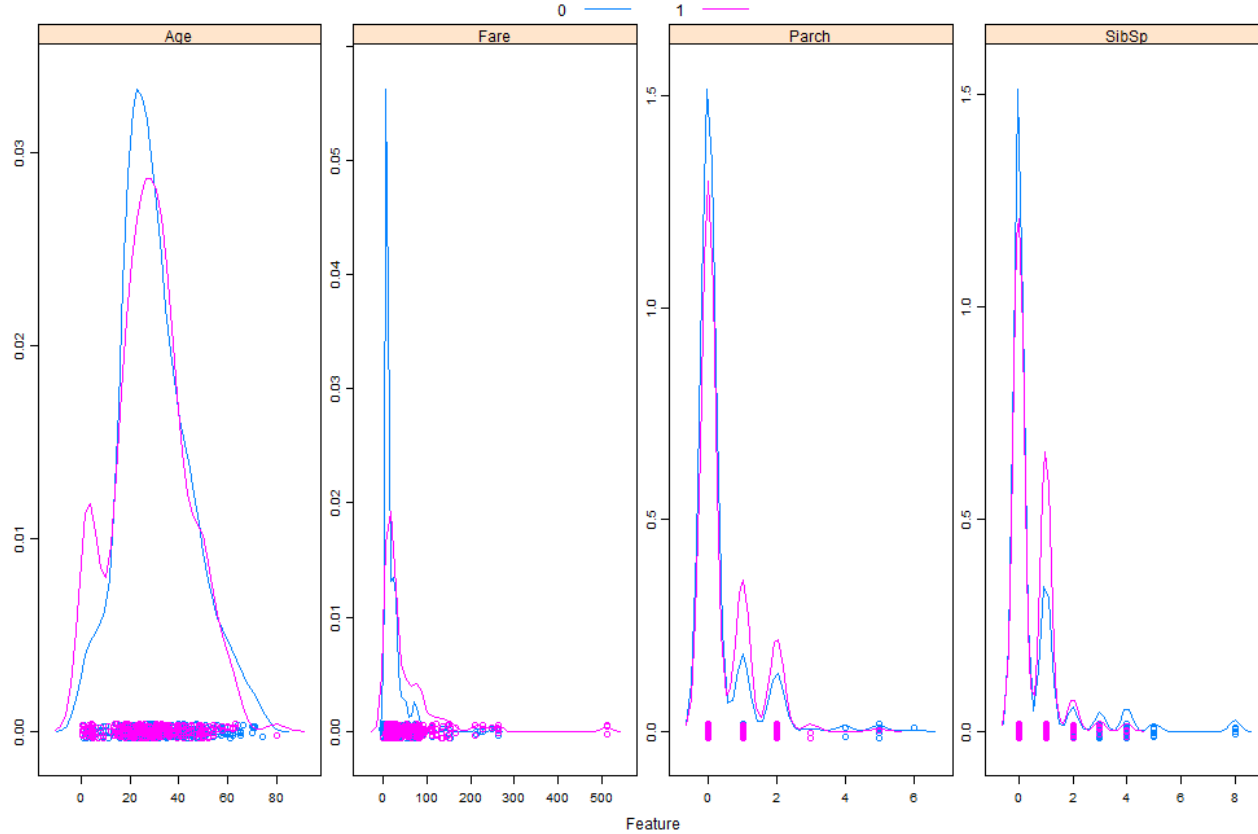
Scatter Plot Matrix

# Caret





# Caret



# Caret

```
set.seed(1234)
inTraining <- createDataPartition(dados$Survived, p=0.6,
                                   list=FALSE)
training.set <- dados[inTraining,]
Totalvalidation.set <- dados[-inTraining,]
invalidation <- createDataPartition(Totalvalidation.set$Survived,
                                     p=0.5, list=FALSE)
testing.set <- Totalvalidation.set[invalidation,]
validation.set <- Totalvalidation.set[-invalidation,]
```

Amostragem balanceada tentando manter a mesma proporção de bons e maus nas diferentes bases.

# Caret

Criação de dummies:

```
299 dummies <- dummyVars(survived ~ sex, data = dados)
300 head(predict(dummies, newdata = dados))
301
```

301:1  (Top Level) ▾

**Console** C:/Users/Cacá Relvas/Desktop/Data Mining Março 2016/ 


```
> dummies <- dummyVars(survived ~ sex, data = dados)
> head(predict(dummies, newdata = dados))
```


	sexfemale	sexmale
1	0	1
2	1	0
3	1	0
4	1	0
5	0	1
6	0	1

# Caret

## Zero and Near Zero-Variance Predictors:

```
302 nearZeroVar(dados, saveMetrics= TRUE)
303
```

303:1  (Top Level) ⬆

**Console** C:/Users/Cacá Relvas/Desktop/Data Mining Março 2016/ 

```
> nearZeroVar(dados, saveMetrics= TRUE)
```

	freqRatio	percentUnique	zeroVar	nzv
PassengerId	1.000000	100.0000000	FALSE	FALSE
Survived	1.605263	0.2244669	FALSE	FALSE
Pclass	2.273148	0.3367003	FALSE	FALSE
Name	1.000000	100.0000000	FALSE	FALSE
Sex	1.837580	0.2244669	FALSE	FALSE
Age	1.111111	9.8765432	FALSE	FALSE
Sibsp	2.909091	0.7856341	FALSE	FALSE
Parch	5.745763	0.7856341	FALSE	FALSE
Ticket	1.000000	76.4309764	FALSE	FALSE
Fare	1.023810	27.8338945	FALSE	FALSE
Cabin	171.750000	16.6105499	FALSE	FALSE
Embarked	3.833333	0.4489338	FALSE	FALSE

# Caret

Filtrar variáveis correlacionadas:

```
304 descrCor <- cor(dados[, c("Age", "SibSp", "Parch", "Fare")],
305                  use = "complete.obs")
306 descrCor
307 findCorrelation(descrCor, cutoff = .75)
```

302:38  (Top Level) ⌵

**Console** C:/Users/Cacá Relvas/Desktop/Data Mining Março 2016/ ➔

```
> descrCor <- cor(dados[, c("Age", "SibSp", "Parch", "Fare")],
+                use = "complete.obs")
```

```
> descrCor
```

	Age	SibSp	Parch	Fare
Age	1.00000000	-0.3082468	-0.1891193	0.09606669
SibSp	-0.30824676	1.00000000	0.3838199	0.13832879
Parch	-0.18911926	0.3838199	1.00000000	0.20511888
Fare	0.09606669	0.1383288	0.2051189	1.00000000

```
> findCorrelation(descrCor, cutoff = .75)
integer(0)
```

# Caret

Centering and scaling:

```
310 preProcValues <- preProcess(dados[, c("Age", "SibSp", "Parch", "Fare")],
311                               method = c("center", "scale"))
312 Transformed <- predict(preProcValues,
313                          dados[, c("Age", "SibSp", "Parch", "Fare")])
314
```

305:39  (Top Level) ▾

**Console** C:/Users/Cacá Relvas/Desktop/Data Mining Março 2016/ ↗

```
> preProcValues <- preProcess(dados[, c("Age", "SibSp", "Parch", "Fare")],
+                               method = c("center", "scale"))
> Transformed <- predict(preProcValues,
+                          dados[, c("Age", "SibSp", "Parch", "Fare")])
+ .
```

# Caret

Imputation:

```
315 aux = dados[, c("Age", "SibSp", "Parch", "Fare")]
316 preProcValues <- preProcess(aux,
317                             method = c("medianImpute"))
318 aux[6,]
319 predict(preProcValues, aux[6,])
320
```

312:35  (Top Level) ⇅

**Console** C:/Users/Cacá Relvas/Desktop/Data Mining Março 2016/ ↗

```
> aux = dados[, c("Age", "SibSp", "Parch", "Fare")]
> preProcValues <- preProcess(aux,
+                             method = c("medianImpute"))
> aux[6,]
  Age SibSp Parch  Fare
6  NA     0     0 8.4583
> predict(preProcValues, aux[6,])
  Age SibSp Parch  Fare
6  28     0     0 8.4583
```

Outras possibilidades: knnImpute, bagImpute

# Caret

Há vários métodos implantados de feature importance:

- Linear: baseado na estatística t.
- Random forest: baseado no erro OOB permutando cada variável.
- Árvore: redução da função de erro em cada quebra.
- Bagged Trees: mesma metodologia da árvore, utilizando a média de todas as árvores.
- Partial Least Square, Boosted trees, MARS, Cubist, Nearest shrunken centroids.



# Caret

```
321 aux <- training.set[, c("Age", "Sibsp", "Parch", "Fare")]
322 preProcValues <- preProcess(aux, method = c("medianImpute"))
323 aux_imp <- predict(preProcValues, newdata=aux)
324 rf <- train(y=as.factor(training.set$Survived),
325            x=aux_imp,
326            method = "rf", trControl = trainControl(method = "oob"),
327            importance = TRUE, verbose = TRUE,
328            tuneGrid = data.frame(mtry = 3))
329 varImp(rf)
330
```

319:25 ⓘ (Top Level) ↕

Console C:/Users/Cacá Relvas/Desktop/Data Mining Março 2016/ ↗

```
> aux <- training.set[, c("Age", "Sibsp", "Parch", "Fare")]
> preProcValues <- preProcess(aux, method = c("medianImpute"))
> aux_imp <- predict(preProcValues, newdata=aux)
> rf <- train(y=as.factor(training.set$Survived),
+           x=aux_imp,
+           method = "rf", trControl = trainControl(method = "oob"),
+           importance = TRUE, verbose = TRUE,
+           tuneGrid = data.frame(mtry = 3))
> varImp(rf)
rf variable importance
```

	Importance
Fare	100.00
Sibsp	45.11
Age	38.28
Parch	0.00

# Caret

Há diversas métricas para medir performance dos modelos implementadas.

Classificação:

- Sensibilidade, sensibilidade, acurácia, matriz confusão, AUC, etc.

Regressão:

- MSE, RMSE, R2, erro absoluto.

# Caret

```
331 aux_val <- validation.set[, c("Age", "SibSp", "Parch", "Fare")]
332 pre_val <- predict(preProcValues, newdata=aux_val)
333 pred_var <- predict(rf, newdata = pre_val)
334 confusionMatrix(pred_var, validation.set$Survived)
335
```

331:43 (Top Level) ↕

Console C:/Users/Cacá Relvas/Desktop/Data Mining Março 2016/ ↗

```
> aux_val <- validation.set[, c("Age", "SibSp", "Parch", "Fare")]
> pre_val <- predict(preProcValues, newdata=aux_val)
> pred_var <- predict(rf, newdata = pre_val)
> confusionMatrix(pred_var, validation.set$Survived)
```

Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	78	35
1	29	36

Accuracy : 0.6404  
95% CI : (0.5653, 0.7109)  
No Information Rate : 0.6011  
P-Value [Acc > NIR] : 0.1599

Kappa : 0.2394  
McNemar's Test P-Value : 0.5320

Sensitivity : 0.7290  
Specificity : 0.5070  
Pos Pred Value : 0.6903  
Neg Pred Value : 0.5538  
Prevalence : 0.6011  
Detection Rate : 0.4382  
Detection Prevalence : 0.6348  
Balanced Accuracy : 0.6180

# Caret

Baseada no seguinte algoritmo:

```
1 Define sets of model parameter values to evaluate
2 for each parameter set do
3   for each resampling iteration do
4     Hold-out specific samples
5     [Optional] Pre-process the data
6     Fit the model on the remainder
7     Predict the hold-out samples
8   end
9   Calculate the average performance across hold-out predictions
10 end
11 Determine the optimal parameter set
12 Fit the final model to all the training data using the optimal parameter set
```

Logo, temos que escolher o tipo de modelo, o grid de parâmetros para serem escolhidos (model tuning) e a estratégia de reamostragem para a otimização dos parâmetros.

# Caret

Algumas estratégias de reamostragem:

- "boot": bootstrap (a performance é a média das performances de todas as bases)
- "cv": cross-validation
- "repeatedcv": repetição da opção de cross-validation várias vezes (a média de todas as performances é utilizada).
- "oob": out of bag. Utilizada em apenas alguns métodos como o random forest, em que o erro out of bag é utilizado para a escolha do melhor grid de parâmetros.

# Laboratório

# Titanic

Exemplos:

Fazer processo completo de modelagem.



Knowledge • 3,464 teams

## Titanic: Machine Learning from Disaster

Fri 28 Sep 2012

Sat 31 Dec

Dashboard

Home



Data



Make a submission



Competition Details » [Get the Data](#) » [Make a submission](#)

## Predict survival on the Titanic using

# Titanic

## VARIABLE DESCRIPTIONS:

survival	Survival (0 = No; 1 = Yes)
pclass	Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
name	Name
sex	Sex
age	Age
sibsp	Number of Siblings/Spouses Aboard
parch	Number of Parents/Children Aboard
ticket	Ticket Number
fare	Passenger Fare
cabin	Cabin
embarked	Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)



# Caret

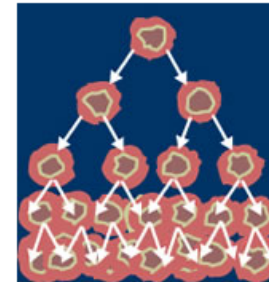
**Exercícios:** <http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29>



## Breast Cancer Wisconsin (Diagnostic) Data Set

Download: [Data Folder](#), [Data Set Description](#)

Abstract: Diagnostic Wisconsin Breast Cancer Database



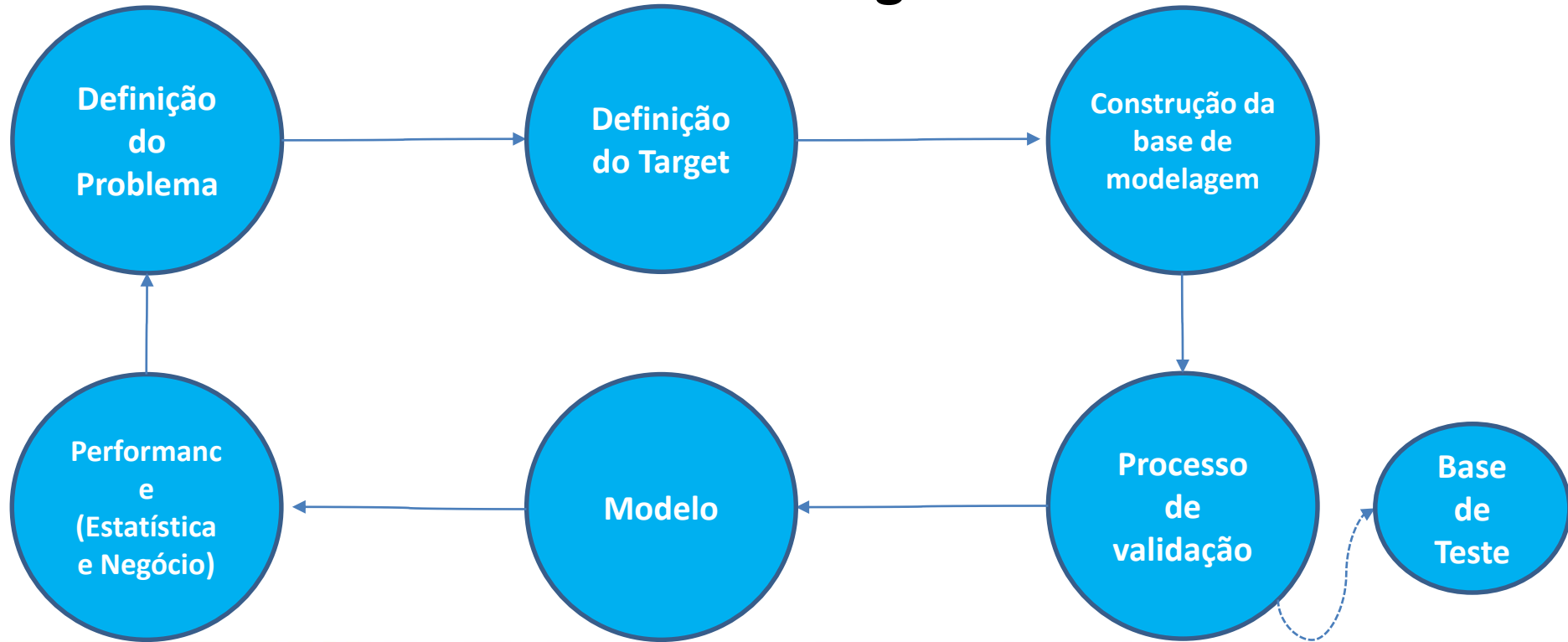
Data Set Characteristics:	Multivariate	Number of Instances:	569	Area:	Life
Attribute Characteristics:	Real	Number of Attributes:	32	Date Donated	1995-11-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	390512

# Caret

- 1.) Utilize seed de 42 e crie amostras de treino (70%) e teste (30%).
- 2.) Faça a análise completa do ajuste.
- 3.) Ajuste os hiperparâmetros.

# Processo de Machine Learning

# Etapas de um projeto de machine learning



# Definição do problema

Temos que definir exatamente o que queremos solucionar. Ex:

- Medir o risco de inadimplência de um cliente.
- Prever os gastos mensais para a definição de um limite ótimo.
- Informar ao atendente qual o possível problema que o cliente que está ligando irá falar a respeito.

Trabalho deve ser feito em conjunto com o analista que irá usar o modelo.

# Definição do target

- Com o problema que queremos solucionar definido, temos a etapa de definição da variável resposta. Extremamente importante no processo!
- Ex.:
  - O que representa o risco de inadimplência? Não pagamento parece ok, mas em quanto tempo? 10 dias de atraso é ruim? Quantos meses de observação vou olhar?
  - O que uso para prever os gastos no próximo mês? O gasto do mês anterior? E efeitos sazonais?

# Construção da base de modelagem

- Usamos todos os dados (variáveis) disponíveis? Será que o signo da pessoa importa? Ou o nome?
- Todas as variáveis devem estar disponíveis no momento da tomada de decisão. Ex: um modelo que irá decidir a aprovação ou não de um cliente só poderá usar variáveis disponíveis naquele momento. Não importa se a renda atual é R\$5.000,00 se no momento da aprovação inicial a renda era R\$2.000,00.
- Uma arquitetura / modelagem de banco de dados apropriada ajuda a garantir isso (histórico).

# Processo de validação

Uma das etapas mais importantes. É graças a esta etapa que conseguimos testar o efeito do modelo na prática. Será que o nosso modelo terá o mesmo comportamento com novos dados?

Há diferentes estratégias de validação:

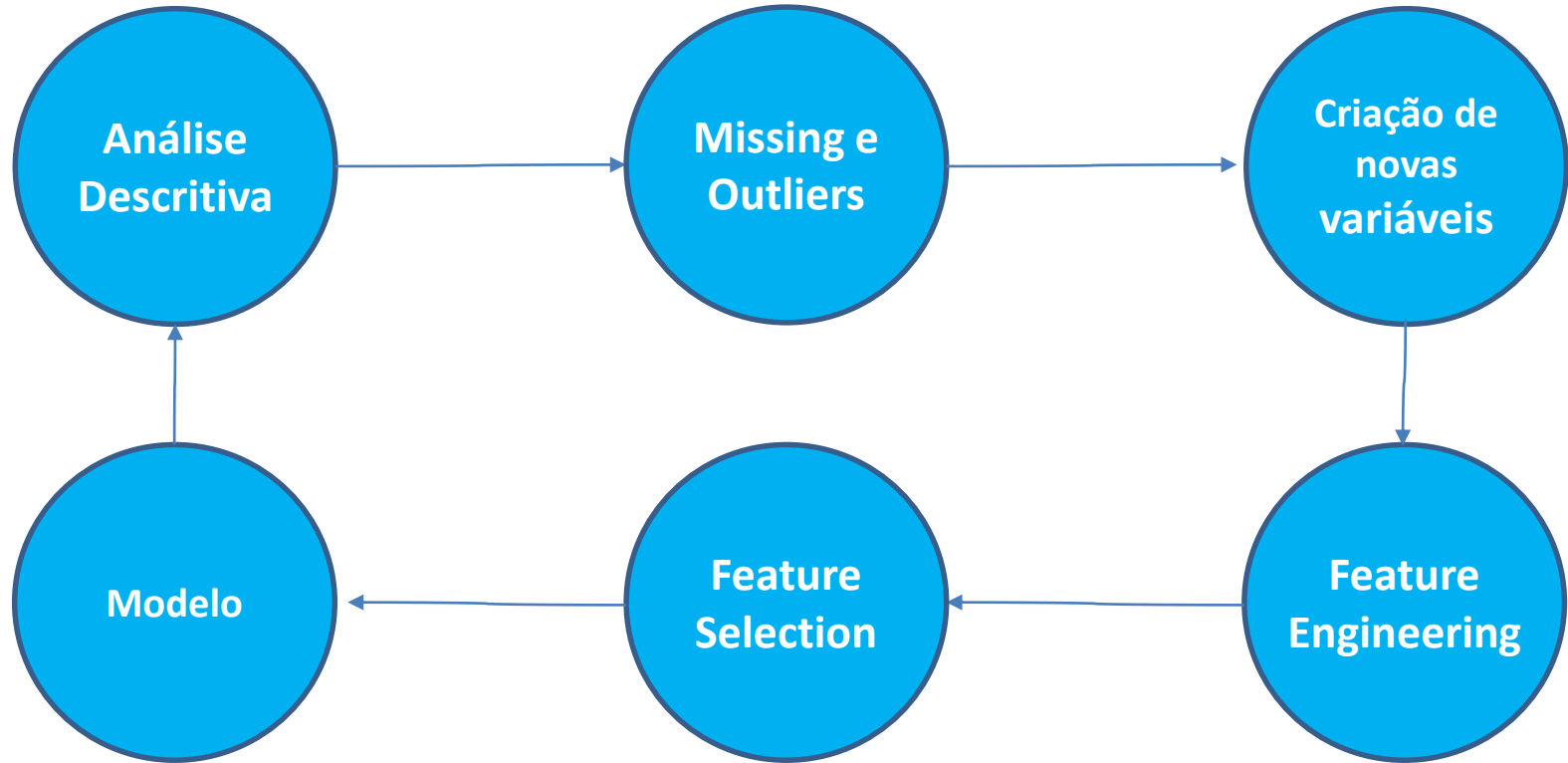
- Base de treino e teste.
- Base de treino, validação e teste.
- Base out-of-time. Útil para checar se o modelo será estável no tempo.
- Validação cruzada.



# Performance (Estatística e Negócio)

- É sempre sugerido definir inicialmente qual métrica estatística será utilizada para verificar a performance do modelo.
- Da mesma forma as métricas de negócio. O modelo está conseguindo solucionar nosso problema?
- Trabalho em conjunto com o analista que irá desenvolver o modelo.
- Padronização para as próximas versões do modelo.

# Etapas do Modelo



# Distribuição das variáveis

- Distribuição e correlação de cada variável com o target.
- Nesta etapa checamos a quantidade de valores missing e possíveis valores aberrantes.
- Obter *insights* para novas variáveis, bem como na melhor forma de tratar a variável com a resposta (é linear, quadrática?).
- Bugs no processo de construção dos dados também são encontrados aqui.

# Missing e Outliers

- Como procedermos com valores missing (ex: nulo, strings vazias)? Algumas técnicas de machine learning são capazes de lidar com isso automaticamente, mas muitas não. Qual a melhor forma de fazer isso?
- Outliers podem impactar significativamente o modelo. Será que temos que fazer algo? É um risco em produção?

# Criação de novas variáveis

- Com as análises anteriores ou conhecimento do negócio, podemos concluir que duas variáveis podem ser combinadas ou que uma transformação é mais apropriada.
- Muitas pessoas tendem a ignorar esta etapa, embora possa produzir grandes melhorias no modelo.

# Feature Engineering

- Qual a melhor forma de tratar cada variável?
- Categorizar é uma boa alternativa?
- Devo usá-la de forma linear?
- Realizar uma transformação não-linear? Ex: spline.

# Feature Selection

- Tenho centenas de variáveis. Devo usar todas? Quais são as mais importantes?
- Há algoritmos que se beneficiam de mais variáveis, outros algoritmos podem apresentar problemas (ex: instabilidade, multicolinearidade, tempo de execução, memória).
- Qual a melhor abordagem para escolher o melhor subconjunto de variáveis?

# Modelo

- Qual técnica de modelagem (ex: Linear, Random Forest, Boosting, Redes Neurais)?
- Trade-off: preciso interpretar os meus resultados (ex: linear), ou posso usar um algoritmo mais poderoso (ex: rede neural)?
- Há alguma restrição regulatória? Ex.: O banco central exige alguma interpretação?



# Modelo

## Missing

Imputação pela média  
/mediana  
Remove  
Imputação por modelo  
Multiple Imputation  
Categorizar

## Outliers

Remove  
Truncar por valores  
mínimos e máximos  
Trocar por um modelo  
Categorizar

## Feature Engineering

Categorização  
Splines  
Tranformações

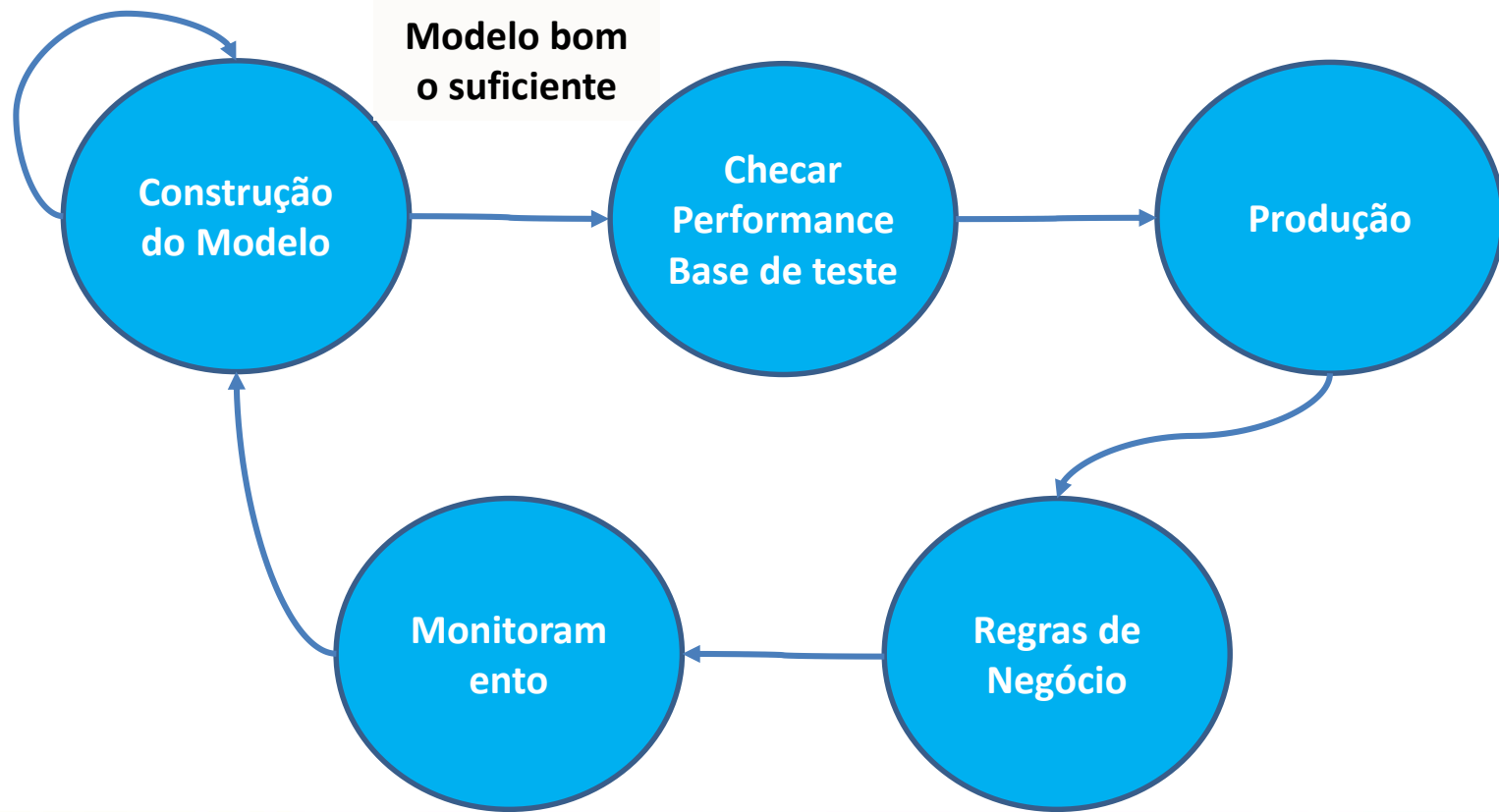
Stepwise  
Random Forest – Importance  
Componentes Principais / Análise  
Fatorial  
Análise de Correlação

## Algoritmos

Regressão Linear / Logística  
Random Forest  
Boosting  
Bagging  
Redes Neurais

## Feature Selection

# Etapas de Implementação



# Referências Bibliográficas

- Hastie, T., Tibshirani, R. & Friedman, J.H. (2001) “The Elements of Statistical Learning”
- Bishop, C.M. (2007) “Pattern Recognition and Machine Learning”
- Mitchell, T.M. (1997) “Machine Learning”
- Abu-Mostafa, Y., Magdon-Ismail, M., Lin, H.T (2012) “Learning from data”
- Theodoridis, S., Koutroumbas, K., (2008) “Pattern Recognition”
- Kuhn, M., Johnson, K., (2013) “Applied Predictive Modeling”
- Burns, P. (2011) “The R inferno”