VBA projet

SIMPLE PERCEPTRON / AR-GARCH MODEL / MLP

CABRAL Alexis / CHENG Eva

21/12/2020

Introduction

Pour la réalisation de ce travail, nous avons essayé de tous automatiser et de rendre l'utilisation de notre application VBA beaucoup plus attractif pour l'utilisateur.
Ainsi tous les boutons présent au sein de notre VBA sont utilisables et fonctionnels et l'utilisateur n'as pas besoin de rentrer des valeurs au sein des cellules même d'Excel.
Notre Code est entièrement commentée et le nom des variables choisi est pertinents.

SIMPLE PERCEPTRON

First Step : Implémentation of the perceptron

L'objectif de la partie sur l'implémentation du perceptron c'est de prédire le sens de variation des rendements d'un actifs à travers un algorithme de perceptron. Pour ce faire on a créé un module de fonction contenants notre code pour la data préparation.

La première étape , c'est de créer la variable à expliquer. La variable à expliquer corresponds au sens de variation des rendements future au jour le jour de l'actifs. L'encodage retenu corresponds à 1 si le taux de rendements au jour suivant est positif et 0 sinon. C'est le contenu de ma variable Target.

```
'Computes the forward rates
Sub forwardRate(ByVal Pt As Double, ByVal Pt1 As Double, ByRef result As Double)
If Pt <> 0 Then
    result = Pt1 / Pt - 1
Else
    Err.Raise 1056, , "Division par zéro dans la fonction forwardRate"
End If
End Sub
```

Cette fonction prends ainsi en parametre deux prix : A savoir le prix de l'actif à la date t et le prix de l'actif à la date t+1.Et retournera le taux de rendement en t+1 de l'actif.

Calcule de la variable à expliquer : Cette variable à expliquer prends des valeurs zéro ou un.
0 si le rendements de l'actif le jour suivants est négatif et 1 sinon. C'est ce qui a été codée dans la fonction ComputeTarget.

```
'Computes the target (vector of forward rate signs)
Sub computeTarget(ByVal prices As Variant, ByRef result As Variant)
Dim forwardReturn As Double
Dim size As Long
Dim counter As Long
size = UBound(prices) - LBound(prices)
ReDim result(LBound(prices) To UBound(prices) - 2)
For counter = LBound(prices) + 1 To UBound(prices) - 1
    Call forwardRate(prices(counter), prices(counter + 1), forwardReturn)
    If forwardReturn >= 0 Then
       result(counter - 1) = 1
    Else
       result(counter - 1) = 0
    End If
Next counter
End Sub
```

Ma fonction prends en parametre les prix de l'actifs et retourne le signe du taux de rendements Forward..

La deuxième étape consiste au calcul des variables explicatives du modèle. Trois variable ont été retenu :

Première Variable :

Moyenne mobile 20 jours des rendements de l'actifs
N.B : On ne peut pas utiliser la moyenne mobile des prix car cela créer un effet d'échelle.De sorte à ce que la valeur de la fonction d'activation sera très proche de 1.Ce qui empêche le calcul du J (Log(1-a) n'existe pas quand a est très proche de 1))

```
'Computes the Moving Average 20 vector
Sub computeMovingAverage20(ByVal returns As Variant, ByRef result As Variant)
Dim counter1 As Long
Dim counter2 As Long
For counter1 = LBound(returns) + 19 To UBound(returns) - 1
    If counter1 = LBound(returns) + 19 Then
       ReDim result(LBound(returns) To LBound(returns))
    Else
       ReDim Preserve result(LBound(result) To UBound(result) + 1)
    End If
    result(UBound(result)) = 0
    For counter2 = counter1 To counter1 - 19 Step -1
       result(UBound(result)) = result(UBound(result)) + returns(counter2)
    Next counter2
    result(UBound(result)) = result(UBound(result)) / 20
Next counter1
End Sub
```

Cette fonction prends en valeur les rendements de l'actifs et calcule la moyenne mobile sur 20 jours de ces rendements.

date	symbol	close	Returns	
01/04/2010	GOOG	283,3	0,002962299	1
05/04/2010	GOOG	284,4	0,003885327	1
06/04/2010	GOOG	283	-0,004886008	0
07/04/2010	GOOG	280,7	-0,008236269	0
08/04/2010	GOOG	282,7	0,007009247	1
09/04/2010	GOOG	282,1	-0,002237926	0
12/04/2010	GOOG	285,3	0,011497347	1
13/04/2010	GOOG	292,3	0,024514129	1
14/04/2010	GOOG	293,4	0,003800479	1
15/04/2010	GOOG	296,5	0,010696119	1
16/04/2010	GOOG	274	-0,075844187	0
19/04/2010	GOOG	274	-9,08676E-05	0
20/04/2010	GOOG	276,5	0,008980215	1
21/04/2010	GOOG	276,1	-0,001333249	0
22/04/2010	GOOG	272,5	-0,013061497	0
23/04/2010	GOOG	271,5	-0,003783836	0
26/04/2010	GOOG	264,8	-0,024495904	0
27/04/2010	GOOG	263,5	-0,004852924	0
28/04/2010	GOOG	263,6	0,000245724	1
29/04/2010	GOOG	265	0,005309987	1
30/04/2010	GOOG	261,9	-0,011842134	0
03/05/2010	GOOG	264,3	0,009320928	1
04/05/2010	GOOG	252,2	-0,045665242	0
05/05/2010	GOOG	253,9	0,006694666	1
06/05/2010	GOOG	248,4	-0,021755329	0

1 Moyenne mobile 20 jour

= Somme des rendements des 20 premier jours/20

2 Moyenne mobile 20 jour

= Somme des rendements des 20 premier jours à l'exclusion du premier jours jusqu'au 21eme jour/20

Deuxième variable : Correspondant **à La bande inferieur de Bollinger :**

Moyenne mobile 20 jours -2 * Ecarttype(Moyenne mobile 20 jours)

Troisième variable : Correspondant **La bande supérieur de Bollinger :**

Moyenne mobile 20 jours +2 * Ecarttype(Moyenne mobile 20 jours)

Nous avons ainsi créer une fonction qui me nous permet de calculer la bande inferieur de Bollinger et la bande supérieur de Bollinger correspondant à la deuxième et à la troisième variable au sein de computeBollingerBands :

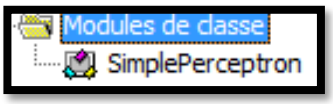
```
'Computes the Lower and Upper Bollinger Band
Sub computeBollingerBands(ByVal returns As Variant, ByRef result As Variant)
Dim size As Long
Dim counter As Long
Dim var_intermedary As Variant
Dim SMA_sd As Double
Call computeMovingAverage20(returns, var_intermedary)
size = UBound(var_intermedary) - LBound(var_intermedary) + 1
If size > 1 Then
   SMA_sd = Application.WorksheetFunction.StDev(var_intermedary)
Else
   SMA_sd = 0
End If
ReDim result(LBound(var_intermedary) To UBound(var_intermedary), LBound(returns) To LBound(returns) + 1)
For counter = LBound(var_intermedary) To UBound(var_intermedary)
    result(counter, LBound(returns)) = var_intermedary(counter) - 2 * SMA_sd
    result(counter, LBound(returns) + 1) = var_intermedary(counter) + 2 * SMA_sd
Next counter
End Sub
```

Afin de répondre à la demande de récupérer les données boursières de Google et les stockez dans un tableau, Nous avons créé une fonction Initialization qui récupère les prix des actifs dans la feuille de donnée et qui les range dans un vecteur VBA.

```
'Recup the prices of the asset in the sheet
Public Sub Initialization(ByRef sheet As Worksheet, ByRef prices As Variant)
Dim counter As Long
counter = 1
While (Not IsEmpty(sheet.Range("A" & counter + 1).value))
    If counter + 1 = 2 Then
       ReDim prices(1 To 1)
    Else
       ReDim Preserve prices(1 To UBound(prices) + 1)
    End If
    prices(UBound(prices)) = sheet.Range("C" & counter + 1).value
    counter = counter + 1
Wend
End Sub
```

Second Step: Implémentation du Model.

Dans cette partie nous avons implémenté un modèle simple de perceptron à travers la création d'un module de classe appelé SimplePerceptron



Ce perceptron utilise par default la sigmoïde comme fonction d'activation.
Ce module sert à créer des instances permettant d'implémenter l'algorithme du perceptron simple.
A l'intérieur de ce module nous avons ajouté des propriétés ainsi que des méthodes. Parmi les méthodes nous avons :

1ere methode : Methode initialization :

```
'Initialization function. This function assignes the inital values to the properties of the class
Public Sub Initialization(ByVal size As Long, ParamArray initialWeigths() As Variant)
   Dim counter As Long
   'If size < UBound(initialWeigths) - LBound(initialWeigths) + 1 Then
       'MsgBox "There more weigths in the paraters than the size of the weigths vector. Only first elements will be used"
   'ElseIf size > UBound(initialWeigths) - LBound(initialWeigths) + 1 Then
       'MsgBox "There less weigths in the paraters than the size of the weigths vector. The reste will be completed with
   'End If
   If size > 0 Then
      ReDim Weigths(1 To size)
   Else
      Err.Raise 1057, , "Error on the size of the weiths vector"
   End If
   If UBound(initialWeigths) > -1 Then
        For counter = LBound(initialWeigths) To Application.WorksheetFunction.Min(size, UBound(initialWeigths))
            Weigths(counter + 1 - LBound(initialWeigths)) = initialWeigths(counter)
        Next counter
        If size > UBound(initialWeigths) Then
            For counter = UBound(initialWeigths) + 1 To size
                Weigths(counter) = Rnd
            Next counter
        End If
   Else
        For counter = 1 To size
            Weigths(counter) = Rnd
        Next counter
   End If
   rate = Feuil5.Range("B2").value
   number_Iterations = Feuil5.Range("B3").value
End Sub
```

Cette méthodes sert à initialiser les valeurs de la propriété Weight de la classe. Cette initialisation se fait de manière aléatoire(Intervalle [0 ; 1]).

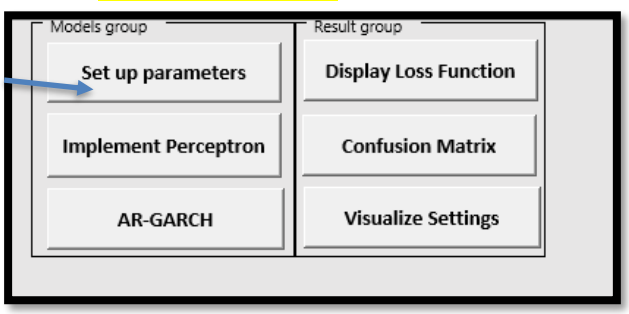
2ere méthode : La méthode Learn :

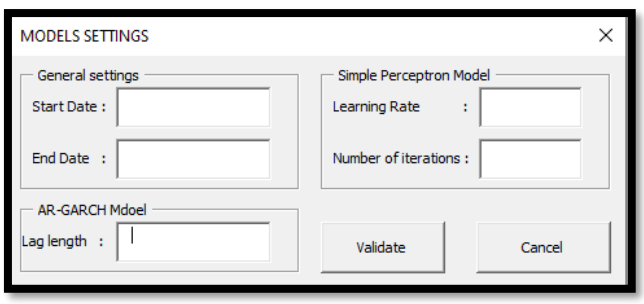
C'est la méthode qui permet d'entrainer le modèle (estimation des parametre du modèle) basée sur l'algorithme 1 donné dans le sujet.

```
'Implements the Gradient descent algorithm for loss minimization of the perceptron
Public Sub Learn(ByVal Y As Variant, ByVal X As Variant)
Dim z As Double
Dim db As Double
Dim dw As Variant
Dim dz As Double
Dim a As Double
Dim J As Double
Dim counter1 As Long
Dim counter2 As Long
Dim result As Variant
db = 0
dw = 0
dz = 0
b = 0
ReDim dw(LBound(Y) To UBound(Y)) As Double
ReDim Bias_Value(1 To number_Iterations) As Double
ReDim Bias_Value(1 To number_Iterations) As Double
ReDim loss_Function(1 To number_Iterations) As Double
For counter2 = LBound(Y) To UBound(Y)
    dw(counter2) = 0
Next counter2
For counter1 = 1 To number_Iterations
   For counter2 = LBound(Y) To UBound(Y)
      z = computeDotProduct(Weigths, recupRow(X, counter2)) + b
      a = sigmaZ(z)
      J = -(Y(counter2) * Log(a) + (1 - Y(counter2)) * Log(1 - a))
      dz = a - Y(counter2)
      dw = vectorsAddition(dw, computeVectorScalarProduct(recupRow(X, counter2), dz))
      db = db + dz
   Next counter2
   J = J / (UBound(Y) - LBound(Y) + 1)
   dw = computeVectorScalarProduct(dw, 1 / (UBound(Y) - LBound(Y) + 1)) 'dw / m
   db = db / (UBound(Y) - LBound(Y) + 1)
```

```
   Weigths = vectorsAddition(Weigths, computeVectorScalarProduct(dw, -rate)) 'w - rate * dw
   b = b - rate * db
   loss_Function(counter1) = J
   Bias_Value(counter1) = b
Next counter1
End Sub
```

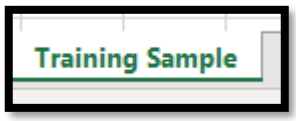
Cette méthode estime les poids du modèle ainsi que le biais et la fonction de perte du modèle.
Pour l'initialisation des paramètres start date , end date Learning rate, Numbers of itération and lag lenght nous avons créé un formulaire qui permet d'entrer les valeurs souhaités. Ce formulaire peut être appelé à travers le bouton Set up parameters de la feuille menu de l'application :





Pour le choix du training Simple et test Sample, l'utilisateur choisi les dates (start date et End date) correspondants à la fenêtre des données pour le training Sample.Le reste des données à partir du end date est utilisée comme test Sample.

Les données du training Sample sont rangé dans la feuille training Sample de l'applications



date	symbol	close	Returns	
01/04/2010	GOOG	283,3	0,002962299	1
05/04/2010	GOOG	284,4	0,003885327	1
06/04/2010	GOOG	283	-0,004886008	0
07/04/2010	GOOG	280,7	-0,008236269	0
08/04/2010	GOOG	282,7	0,007009247	1
09/04/2010	GOOG	282,1	-0,002237926	0
12/04/2010	GOOG	285,3	0,011497347	1
13/04/2010	GOOG	292,3	0,024514129	1
14/04/2010	GOOG	293,4	0,003800479	1

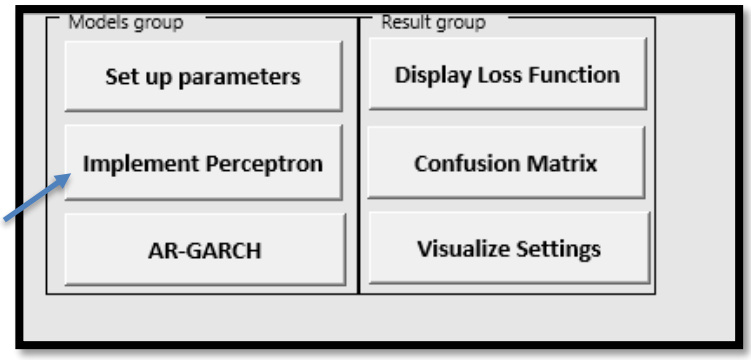
Les données du test Sample sont rangé dans la feuille test Sample de l'applications

date	symbol	close	Returns
31/10/2016	GOOG	784,539978	-0,01361633
01/11/2016	GOOG	783,609985	-0,0011854
02/11/2016	GOOG	768,700012	-0,01902729
03/11/2016	GOOG	762,130005	-0,00854691
04/11/2016	GOOG	762,02002	-0,00014431
07/11/2016	GOOG	782,52002	0,02690218
08/11/2016	GOOG	790,51001	0,01021059

C'est la fonction fillDataSet du module fonctions qui permet de récupérer les données et de les affecter aux feuilles correspondantes.

```
'Split the data into trainning sample and test sample in input them on the dedicated worksheets
Sub fillDataSet()
Dim initialPos As Long
Dim endPos As Long
Dim startD As Date
Dim endD As Date
startD = ThisWorkbook.Worksheets("Settings").Range("B9")
endD = ThisWorkbook.Worksheets("Settings").Range("B10")
initialPos = seekStartDatePosition(startD)
endPos = seekEndDatePosition(endD)
If initialPos > 0 Then
   Call eraseColumn(ThisWorkbook.Worksheets("Training Sample"), 1)
   Call eraseColumn(ThisWorkbook.Worksheets("Training Sample"), 2)
   Call eraseColumn(ThisWorkbook.Worksheets("Training Sample"), 3)
   Call eraseColumn(ThisWorkbook.Worksheets("Training Sample"), 4)
   ThisWorkbook.Worksheets("Data").Range("A" & initialPos & ":D" & endPos).Copy
   ThisWorkbook.Worksheets("Training Sample").Range("A2").PasteSpecial xlPasteValues
   Call eraseColumn(ThisWorkbook.Worksheets("Test Sample"), 1)
   Call eraseColumn(ThisWorkbook.Worksheets("Test Sample"), 2)
   Call eraseColumn(ThisWorkbook.Worksheets("Test Sample"), 3)
   Call eraseColumn(ThisWorkbook.Worksheets("Test Sample"), 4)
   initialPos = endPos + 1
   endPos = ThisWorkbook.Worksheets("Data").Range("A1").End(xlDown).Row
   If (endPos >= initialPos) Then
      ThisWorkbook.Worksheets("Data").Range("A" & initialPos & ":D" & endPos).Copy
      ThisWorkbook.Worksheets("Test Sample").Range("A2").PasteSpecial xlPasteValues
   End If
End If
End Sub
```

Dans la fonction main du menu main nous faisons appels aux différentes fonctions permettant à la fois de créer, d'entrainer et de prédire les valeurs du Target à travers la classe simple perceptron.(En cliquant sur le boutons Implement Perceptron de la feuille menu.



Modele du perceptron et son interprétation :

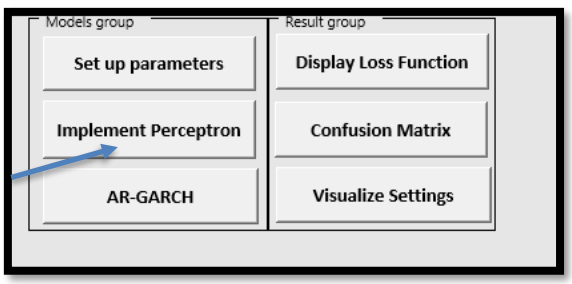
Après avoir rentré les paramètres dans le Set up Parameters et après les avoir charger en cliquant sur Implement Perceptron , il est possible d'obtenir un graphique de la fonction de perte en cliquant sur Display loss function.
Apres plusieurs tests effectuée, nous avons pu constater que notre fonction converge pour certaines fenêtres des données utilisé pour le training set et diverge pour d'autres fenêtres.
L'algorithme n'est pas donc pas stable car la convergence dépends des données.

Interprétation de la weight value

A titre d'exemple nous avons implémenté l'algorithme avec une fenêtre de training set correspondant à :
Star date : 01/04/2010
End date : 30/10/2016
Learning rate : 0,05
Numbers of iteration : 20

Result Perceptron	
Coeff SMA20	0,962
Coeff Bollinger Lower Band	0,871
Coeff Bollinger Upper Band	0,056
Bias	0,913

Ces 3 variables contribue positivement sur la valeur du rendements futurs de l'actifs.
Toujours au sein de notre exemple, Pour la matrice de confusion elle est disponible dans la fenêtre Prediction.
Matrice photo :



Confusion matrix			
		Real values	
		1	0
Prédicted	1	10	13
	0	0	0

L'algorithme implémenté arrive donc à prédire 10 vraie positif et 13 faux négatifs. La précision est donc de 43,47% mais cela dépends toujours des parametre donc du Learning rate et du nombre d'itérations.

AR-GARCH model

L'objectif du model AR-Garch est de prédire les rendements futurs en fonction de leurs valeur passée ainsi que de leurs volatilité passée.
Pour ce faire nous avons créés une feuille Excel du nom de ARGarch permettant d'implémenter ce modèle.

- La première colonne contient les dates de cotations selon la fenêtre d'entrainements choisie.(Start date et end date)
- La deuxième colonne contient les prix de l'actifs
- La troisième colonne contient les rendements.
- La quatrième colonne contient les prédictions (partie autorégressif du modèle) de rendements selon les paramètres initiaux du modèle

$$y_t = \sum_{i=1}^{m} \eta_i y_{t-i}$$

Pour ce faire nous avons crée une fonctions qui fait la mise à jour de la formule dans les cellules de la colonne D en fonction du nombre n (Numbers of lags)

Cette fonction de mise à jour est dans la procédure Garch Main

```
'Update formulas in the column D to H of the sheet AR-GARCH
Feuil6.Range("D" & nb).FormulaLocal = "=" & formula
rowLim = limitRow(Feuil6, 1, "A")
Feuil6.Range("D" & nb).Copy
Feuil6.Range("D" & nb + 1 & ":D" & rowLim).PasteSpecial Paste:=xlPasteFormulas
```

De même on complète la colonne E qui contient les résidus du modèle AR-Garch.

Résidus = rendements – AR(n)

Au niveau de la feuille cela reviens à faire la cellule C moins la cellule D

La colonne F représente la volatilité conditionnelle du modèle.

La formule est la suivante :

$$\sigma_t^2 = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2.$$

Pour ce faire la fonction garch main se charge de faire la mise à jour, de la formule dans les cellules de la colonne F en fonction de la cellule précédente et de la colonne epsilon.

La colonne G représente Epsilon et epsilon représente les résidus (Bruit blanc)
Les résidus (Bruit blanc) se calcule en divisant les résidus de la partie autorégressif du modèle par la volatilité conditionnelle du modèle.

La colonne H représente le loglikelihood donc la log vraisemblance du modèle.

log vraisemblance du modèle = ½*(ln(pi*2)-2*(ln(volatilité conditionnelle – Ar(n)²/ volatilité conditionnelle²))

Nous l'avons directement mis la formule dans les cellules de la colonne H.
Puis on calcule la log vraisemblance totale du modèle.En faisant la somme des éléments de la colonne H.Ce qui représente le log likehood en résultat de M2

La variance de epsilon permet de valider le modèle en effet à l'optimum la variance du epsilon est très proche de 1 voir égale à 1.

Les paramètres du modèle sont estimées par maximisation de la log vraisemblance globale du modèle (cellule M2 feuille ArGarch).Cette maximisation se fait automatiquement à travers le solver qui est appelé et paramétrer dans la procédure garchMain()

```
'Parametrization of the solver
formula = "$K$4,$K$6,$K$7,$K$8"
For counter = 10 To 9 + n
    formula = formula & ",$K$" & counter
Next counter
SolverOk SetCell:="$M$2", MaxMinVal:=1, ValueOf:=0, ByChange:= _
        formula, Engine:=1, EngineDesc:="GRG Nonlinear"
SolverSolve
Application.ScreenUpdating = True
End Sub
```

Comparaison :

Le premier point : first and last date of Learning sample sont paramétrés à travers le boutons set up parameters de la feuille menu qui fait appelle au formulaire permettant d'entrer les valeur

Le deuxième point : first and last date of test sample sont déduits des données globales en prenants la première date au-delà de la n date du training sample jusqu'à la dernière date (date la plus récente de cotation)

Au sein d'un exemple, Pour la matrice de confusion du AR-Gach model,elle est disponible dans la feuille Argachconfuse .

Confusion matrix			
		Real values	
		1	**0**
Prédicted	**1**	11	9
	0	10	11

L'algorithme arrive à prédire 11 vraie positif et 9 faux négatifs. Et 1 à faux négatifs et 11 vrai négatifs
La précision est donc de 53,66%.