Fil Rouge Documentation

Version 1.0

PAILLEUX Jean-Didier - ZEMNI Malek - KLOTZ Sonny

mai 26, 2017

Table des matières

CHAPITRE 1

gestionFlux.py

1.1 Le module `Gestion des flux`

interface_web.gestionFlux.**fenetre_choix_fichier**()
Fonction qui affiche le template « choix_fichier.html » lorsque la requette HTTP « / » est indiquée. Elle se charge également de vider le dossier "uploads/" avec la fonction removeFiles()

Retourne retourne le template « choix_fichier.html »

interface_web.gestionFlux.**fenetre_role_choix_colonne**(*file*)
Fonction qui affiche le template « role_choix_colonne.html » lorsque la requette HTTP « /fenetre_role_choix_colonne/ » est indiquée. Le template affiche un message d'erreur si le fichier "file" n'est pas valide, sinon elle affiche son contenu.

Paramètres **file** – représente le nom du fichier chargé

Retourne retourne le template « role_choix_colonne.html »

interface_web.gestionFlux.**fenetre_resultat_ADD**(*file*)
Fonction qui affiche le template « resultat_ADD.html » lorsque la requette HTTP « /fenetre_resultat_ADD/ » est indiquée. Cette fonction se charge également défectuer tout les calculs autour de l'analyse descriptives de données avant l'affichage de la page.

Retourne retourne le template « resultat_ADD.html »

interface_web.gestionFlux.**remove**(*file*)
Fonction qui supprime le fichier "file" mis en paramètre qui se situe dans le dossier "uploads/".

Param file de type str correspondant au nom du fichier csv

Retourne redirige vers la route index

interface_web.gestionFlux.**removeFiles**()
Fonction qui se charge simplement de vider le dossier contenant les fichiers chargé.

interface_web.gestionFlux.**manuel**()
Fonction qui affiche le template manuel.html » lorsque la requette HTTP « /manuel/ » est indiquée. Ce template contiend le pdf du manuel utilisateur.

Retourne retourne le template « manuel.html »

interface_web.gestionFlux.**sauvegardeResultats**(*file*)

Fonction qui sauvegarde les résultats de l'analyse descriptives dans un fichier .csv, et lance ainsi son téléchargement.

Retourne téléchargement du fichier "Resultats.csv".

CHAPITRE 2

choixFichier.py

2.1 Le module `Fenêtre choix fichier`

interface_web.choixFichier.**FileWithSGF**()
Fonction qui se charge de l'upload d'un fichier lors du parcours dans le système de gestions de fichiers.

Retourne redirige la page web vers la fenetre_rôle_choix_colonne avec comme parmètre le chemin du fichier uploadé.

interface_web.choixFichier.**FileWithDragDrop**()
Fonction qui se charge de l'upload d'un fichier après avoir déposé ce fichier dans la zone de Drag&Drop.

Retourne redirige la page web vers la fenetre_rôle_choix_colonne avec comme parmètre le chemin du fichier uploadé.

CHAPITRE 3

addRoutes.py

3.1 Fichier `addRoutes.py` contenant les fonctions d'échanges ajax/serveur

interface_web.addRoutes.**iStats**()
Fonction réalisant un échange ajax entre le serveur et la page web pour envoyer les données Statistiques

Return jsonify(data) data sérialisé en dictionnaire, où data contient les informations de "stat.js"

interface_web.addRoutes.**timeSeries**()
Fonction réalisant un échange ajax entre le serveur et la page web pour envoyer les données des séries Temporelles

Return jsonify(data) data sérialisé en dictionnaire, où data contient les informations de "timeSeries.js"

interface_web.addRoutes.**distribution**()

Fonction réalisant un échange ajax entre le serveur et la page web pour envoyer les données de la ditribution

Return jsonify(data) data sérialisé en dictionnaire, où data contient les informations de "distribution.js"

interface_web.addRoutes.**distributionCumulative**()
Fonction réalisant un échange ajax entre le serveur et la page web pour envoyer les données de la ditribution cumulatives

Return jsonify(data) data sérialisé en dictionnaire, où data contient les informations de "distributionCumulative.js"

CHAPITRE 4

addQualitatives.py

4.1 Le module `Analyse de données qualitatives`

add.addQualitatives.**nbElemListeCouple**(*listeEffectifs*)

Calcul l'effectif total des données de listeEffectifs.

La fonction se charge simplement de calculer l'effectif total des données contenu danslisteEffectifs en sommant l'effectif de chaque tuple (couple[1]).

Paramètres **listeEffectifs** – liste de tuples (donnée, effectif)

Retourne l'effectif total des données de listeEffectifs

Type retourné int

add.addQualitatives.**calculEffectifs**(*listeDonnees*)

Calcul l'effectifs pour chaque données contenu dans listeDonnees.

La fonction prend en entrée une liste contenant les données à analyser. Elle calculera les effectifs pour chaque valeur à l'aide d'un dictionnaire. Ce dictionnaire sera converti en liste de tuples et un tri sera effectué pour ordonner les tuples.

Paramètres **listeDonnees** – liste contenant les données é analyser

Retourne listeEffectifs : liste de tuples (donnée, effectif)

Type retourné list

add.addQualitatives.**calculEffectifsCumules**(*listeEffectifs*)

Calcul l'effectifs cumulés avec l'aide de listeEffectifs.

La fonction prend en entrée la liste des effectifs. Elle calculera dans une nouvelle liste l'effectifs cumulés à partir de « listeEffectifs » en remplaçant l'effectif par l'effectif cumulés.

Paramètres **listeEffectifs** – liste de tuples (donnée, effectif)

Retourne listeEffectifsCumules : liste de tuples (donnée, effectif cumulé)

Type retourné list

add.addQualitatives.**calculFrequences**(*listeEffectifs*)

Calcul les fréquences d'apparitions des valeurs.

La fonction prend en entrée la liste des effectifs. Elle calculera dans une nouvelle liste la fréquence à partir de « listeEffectifs » en remplaçant l'effectif par la fréquence.

Paramètres **listeEffectifs** – liste de tuples (donnée, effectif)

Retourne listeFrequences : liste de tuples (donnée, frequence)

Type retourné list

add.addQualitatives.**calculFrequencesCumulees**(*listeEffectifsCumules*)

Calcul les fréquences cumulés pour une liste de fréquences.

La fonction prend en entrée la liste des effectifs cumulés. Elle calculera dans une nouvelle liste la fréquence à partir de « listeEffectifsCumules » en calculant la fréquence cumulée grâce à l'effectif total

Paramètres **listeFrequences** – liste de tuples (donnée, frequence)

Retourne listeFrequences : liste de tuples (donnée, frequence cumulé)

Type retourné list

add.addQualitatives.**infoSecteurs**(*listeFrequences*)

Stock dans un fichier .js les informations nécessaire à la création d'un diagramme de secteurs.

La fonction prend en entrée le résultat du calcul des fréquences .Elle va créer un fichier .js pour y stocker (écrire) les données nécessaires à la construction du diagramme en secteurs.

Format du fihier : Début {

« donnee1 » : frequence1 associee, « donnee2 » : frequence3 associee, . . .

} Fin

Paramètres **listeFrequences** – liste de tuples (donnée, fréquence)

add.addQualitatives.**infoHistogramme**(*listeEffectifs*)

Stock dans un fichier .js les informations nécessaire é la création d'un histogramme.

La fonction prend en entrée le résultat du calcul des effectifs préalablement stocké dans une liste listeEffectifs. Elle va créer un fichier .js pour y stocker les données nécessaires à la construction de l'histogramme.

Format du fihier : Début {

« x » : liste des donnees « value » : liste des effectifs respectifs

} Fin

Paramètres **listeEffectifs** – liste de tuples (donnée, effectif)

CHAPITRE 5

addQuantitativesContinues.py

5.1 Le module `Analyse de données quantitatives continues`

add.addQuantitativesContinues.**discretisation**(*nombreClasses*, *donneesContinues*)

Discrétise des données continues du paramètre.

La fonction se charge de décomposer l'étendue [min ; max] de l'ensemble de données en `nombreClasses` intervalles de même étendue. Ensuite de remplacer les occurrences des données par l'intervalle auquel la donnée appartient.

Paramètres **donneesContinues** – liste de nombres flottants

Retourne liste d'intervalles, et étendue discrétisée

add.addQuantitativesContinues.**calculNombreClasses**(*donneesContinues*)

Calcule le nombre de classes nécessaire à une discrétisation selon la règle de Sturges.

Type retourné int

> **Avertissement :** Si la distribution n'est pas symétrique, le nombre de classes ne sera pas optimal.

add.addQuantitativesContinues.**preparationIntervallesAnalyse**(*listeIntervalles*)

Prépare les données pour l'utilisation des éléments de calcul du module ADD quantitatives discrètes.

Pour effectuer les analyses descriptives dans le cas continu, la démarche est la même (sauf quantiles) que pour le cas discret. On utilisera cependant comme données les centres des intervalles.

Paramètres **listeIntervalles** – liste issue de la discrétisation des valeurs.

Retourne liste de flottants.

add.addQuantitativesContinues.**interpolationLineaire**(*p1*, *p2*, *y*)

Calcule l'abscisse par interpolation linéaire

Les points `p1`, `p2` nous permettent de définir une fonction linéaire f(x) = pente * x + ordonnée à l'origine (oo). On retrouve ensuite l'abscisse du point d'ordonnée `y` se trouvant sur la courbe de la fonction, x = (`y` - oo) / pente.

Retourne abscisse de l'ordonnée `y` par rapport à la droite (`p1`, `p2`)

Type retourné float

add.addQuantitativesContinues.**quantileContinu**(*ordre*, *listeFrequencesCumulees*, *intervalles*)

Calcule les quantiles d'ordre `ordre` pour une analyse de données conitnues.

Le quantile discret nous permet de retrouver le centre de l'intervalle qui contient le vrai quantile. Ensuite, à partir de l'intervalle et de l'ordre, on en déduit une valeur plus précise par interpolation linéaire.

La fonction linéaire est définie à l'aide des bornes de l'intervalle, on a besoin de deux points : L'ordonnée de la borne supérieure est la fréquence cumulée du centre de l'intervalle (1 si borne = max) L'ordonnée de la borne inférieur est la fréquence cumulée du centre de l'intervalle précédent (0 si borne = min)

Retourne le quantile d'ordre `ordre`

add.addQuantitativesContinues.**infoDistributionCumulativeContinue**(*listeEffectifsCumules*, *intervalles*)

Écriture dans le fichier distributionCumulative.json

Format du fihier : Début {

« x » : [liste des abscisses / bornes des intervalles], « value » : [liste des ordonnées / effectifs cumulés]

} Fin

Paramètres **`listeEffectifCumules`** – liste de couples (centre de l'intervalle, effectif cumulé).

CHAPITRE 6

addQuantitativesDiscretes.py

6.1 Le module `Analyse de données quantitatives discrètes`

add.addQuantitativesDiscretes.**moyenne**(*listeEffectifs*)
Calcule la moyenne arithmétique.

Paramètres **`listeEffectifs`** – liste de couples (valeur, occurences)

Retourne moyenne arithmétique des valeurs de la liste

add.addQuantitativesDiscretes.**quantileDiscret**(*ordre*, *listeFrequencesCumulees*)
Calcule le quantile d'ordre `ordre`.
Quantile non défini si l'ordre n'est pas compris entre 0 exclus et 1 exclus

Paramètres
— **`ordre`** – Nombre flottant compris entre 0 et 1.
— **`listeFrequencesCumulees`** – liste de couples (valeur, frequence cumulee) triée selon les valeurs

Retourne La première valeur telle que la fréquence cumulée correspondante soit supérieure ou égale à l'ordre.

Type retourné float

Note : La médiane est le quantile d'ordre 1/2. Les quartiles sont les quantiles d'ordre 1/4 et 3/4.

---

add.addQuantitativesDiscretes.**variance**(*listeEffectifs*)
Calcule la variance.

add.addQuantitativesDiscretes.**ecartType**(*variance*)
Calcule l'écart-type.

add.addQuantitativesDiscretes.**anomaliesTukey**(*listeEffectifs*)
Liste les valeurs aberrantes de la liste.
Une valeur est dite aberrante selon la règle de Tukey si elle n'appartient pas à un intervalle I définit tel que : I = [Q1 - k * IQ ; Q3 + k * IQ] , k constante réelle Q1 et Q3 les quartiles, IQ l'écart inter-quartiles.

La constante k est choisie arbitrairement égale à 1,5. La valeur 1.5 est selon Tukey une valeur pragmatique, qui a une raison probabiliste. Si une variable suit une distribution normale, alors la zone délimitée par la boîte et les moustaches devrait contenir 99,3 % des observations.

Type retourné list

Retourne Collection contenant les données anormales pour la distribution des valeurs.

add.addQuantitativesDiscretes.**symetrie**(*listeEffectifs*)

Calcule le coefficient de symétrie de Fisher.

Si le coefficient est proche 0, la distribution est approximativement symétrique. Si le coefficient est positif, la distribution est étalée sur la droite. Si le coefficient est négatif, la distribution est étalée sur la gauche.

En théorie si l'écart-type est égal à 0, la symétrie n'est pas définie. Cependant un écart-type égal à 0 s'interprète :

Si toutes les valeurs de la distribution sont égales à la moyenne, notre écart-type va être nul. On peut alors considérer la distribution parfaitement symétrique, toutes les données sont regroupées en un point, la moyenne.

Type retourné float

add.addQuantitativesDiscretes.**aplatissement**(*listeEffectifs*)

Calcule le coefficient d'aplatissement de Fisher.

Si le coefficient est égal à 3, la distribution suit une loi normale centrée réduite. Si le coefficient est inférieur à 3, la distribution est aplatie. Si le coefficient est supérieur à 3, les valeurs de la distribution est concentrée autour de la moyenne.

Non défini si l'écart-type est nul

Type retourné float

add.addQuantitativesDiscretes.**infoDistributionDiscrete**(*listeEffectifs*)

Écriture dans le fichier distribution.js

Format du fihier : Début {

« x » : [liste des abscisses], « value » : [liste des ordonnées / effectifs]

} Fin

Paramètres **listeEffectifs** – liste de couples (valeur, effectif).

add.addQuantitativesDiscretes.**infoDistributionCumulativeDiscrete**(*listeEffectifsCumules*)

Écriture dans le fichier distributionCumulative.js

Format du fihier : Début {

« x » : [liste des abscisses], « value » : [liste des ordonnées / effectifs cumulés]

} Fin

Paramètres **listeEffectifCumules** – liste de couples (valeur, effectif cumulé).

add.addQuantitativesDiscretes.**infoBoiteTukey**(*listeEffectifs*)

Écriture dans le fichier boxplot.js

Format du fihier : Début {

« q1 » : premier quartile, « median » : mediane, « q3 » : troisième quartile, « left » : extrémité gauche de la moustache (q1 - 1.5*(q3-q1)), « right » : extrémité droite de la moustache (q3 + 1.5*(q3-q1)), « outliers » : liste des anomalies statistiques

} Fin

Informations utiles à la création d'une boîte à moustaches de Tukey

Paramètres **listeEffectifs** – liste de couples (valeur, effectif).

add.addQuantitativesDiscretes.**infoSerieTemporelle**(*listeSerieTemporelle*)

Écriture dans le fichier timeSeries.js

Format du fihier : Début {

« x » : [liste des Timestamp], « value » : [liste des valeurs]

} Fin

Paramètres **listeSerieTemporelle** – liste de couples (Timestamp, valeur), et un Timestamp est une chaîne de caractères.

CHAPITRE 7

intervalle.py

7.1 Le module `intervalle`

Ce module est utilisé pour une manipulation réduite des intervalles dans R.

Example

>>>from add import intervalle >>>i = Intervalle(0, 1, true ,false) >>>i.contient(0.8) True

L'intervalle i de l'exemple sert à représenter [0, 1[en notation mathématiques.

`add.intervalle.`**`rechercheIntervalle`**(*intervalles*, *nombre*)

Retrouve l'intervalle contenant un nombre en parmaètre

Les intervalles de la liste sont distincts. Les intervalles de la liste forment une partition de l'étendue qu'ils représentent et sont triés. Ainsi, le nombre (on suppose qu'il appartient à l'étendue) appartient à un unique intervalle de la liste.

Exemple : etendue = [1, 3] l = { [0, 1[; [1, 2[; [2, 3] }

Retourne retourne l'intervalle auquel `nombre` appartient, False sinon

CHAPITRE 8

verificationFormatFichier.py

8.1 Le module Vérification format fichier

chargement_des_donnees.verificationFormatFichier.**verifExistence**(*chemin*)
Fonctionnalité de vérification de l'existance du fichier pour l'ouverture

Paramètres **chemin** (*str*) – chemin du fichier

Retourne entier 0 ou une description de l'erreur

chargement_des_donnees.verificationFormatFichier.**verifExtension**(*chemin*)
Fonctionnalité de vérification de l'extension du fichier

Paramètres **chemin** (*str*) – chemin du fichier

Retourne entier 0 ou une description de l'erreur

chargement_des_donnees.verificationFormatFichier.**verifLecture**(*fichierCSV*)
Fonctionnalité de vérification de l'accès au contenu du fichier et de sa nature

Paramètres **fichierCSV** (*TextIoWrapper*) – le fichier CSV ouvert

Retourne entier 0 ou une description de l'erreur

chargement_des_donnees.verificationFormatFichier.**ouvrir**(*chemin*)
Fonctionnalité principale d'ouverture du fichier CSV et de vérification

Paramètres **chemin** (*str*) – chemin du fichier

Retourne le fichier CSV ouvert ou la description de l'erreur rencontrée lors de l'ouverture

CHAPITRE 9

analyseContenuFichier.py

9.1 Le module `Analyse Contenu fichier`

chargement_des_donnees.analyseContenuFichier.**lecture**(*fichierCSV*, *toClose*)
Fonction de lecture du contenu du fichier CSV ligne par ligne

Paramètres **fichierCSV** (*TextIoWrapper*) – fichier CSV ouvert et vérifié

Retourne liste dont chaque élément est une sous-liste contenant les données d'une ligne du fichier

chargement_des_donnees.analyseContenuFichier.**typeDeDonnee**(*chaine*)
Fonction de detection du type de donnée depuis une chaine de carcteres

Paramètres **fichierCSV** (*TextIoWrapper*) – fichier CSV

Retourne liste dont chaque élément est une sous-liste contenant les données d'une ligne du fichier

chargement_des_donnees.analyseContenuFichier.**removeDateSuffix**(*chaineDate*)

chargement_des_donnees.analyseContenuFichier.**descriptionColonnes**(*lignesCSV*)
Fonction de description du nom, du type et des erreurs des colonnes du fichier CSV

Paramètres **lignesCSV** (*list*) – lignes du fichier CSV

Retourne dictionnaire de 3 sous-listes ayant pour clés : « nom », « type » et « erreurs »

chargement_des_donnees.analyseContenuFichier.**analyseFichier**(*fichierCSV*)
Fonctionnalité principale d'analyse du contenu du fichier CSV ouvert

Paramètres **fichierCSV** (*TextIoWrapper*) – le fichier CSV ouvert et vérifié

Retourne une liste contenant les données du fichier et un dictionnaire décrivant ces données

Index des modules Python

a

c

i