**Description du package sdmApp**

1. **Historique**
2. **Structure du package**

Le Package **sdmApp** est constitué d’un ensemble de dossiers, de sous dossiers et des fichiers.

Pour une meilleure compréhension, nous présentons dans les lignes suivantes la description de chaque dossier ainsi que sa composition.

1. **Dossiers**
2. **Le dossier R**

C’est le dossier principal contenant les fichiers de scripts R. Ces fichiers contiennent le code de création des différentes fonctions. Dans ce dossier, on y trouve les fichiers tels que :

1. **sdmApp.R :** contient le code source de la fonction sdmApp
2. **sdmApp\_PA.R :** contient le code source de la fonction sdmApp\_PA
3. **sdmApp\_RasterPlot.R :** contient le code source de la fonction sdmApp\_RasterPlot
4. **sdmApp\_TimesRasters.R :** contient le code source de la fonction sdmApp\_TimesRaster
5. **sdmApp\_fold\_Exporer.R :** contient le code source de la fonction sdmApp\_fold\_Explorer
6. **Le dossier INST**

Il s’agit du dossier "installation" standard dans les packages R. Il contient les ressources qui doivent être installées avec le package (documentation, images, données d'exemple). Ce dossier INST comprend trois sous dossiers répartis comme suit :

* **Docs**

Ce sous dossier comprend plus d’une dizaine de fichiers de format PNG.

Parmi ces fichiers on peut citer l’exemple du logo du package sdmApp. Ce dossier facilite la documentation du package

* **Extdata**

**Extdata** est un sous dossier qui regroupe un ensemble de fichier de donnée de format tif ou sav.

* **Shiny/sdmApp**

Le sous dossier **Shiny/sdmApp** contient à la fois des sous dossiers et des fichiers. Parmi ces sous dossier, on a :

* **Controllers** : contient des scripts d’extension **.R** , représentant un traitement spécifique.
* **WWW :** utilisé pour les ressources statiques dans l’application Shiny. Il contient des fichiers d’extensions **.css** pour le style personnalisé.

Parmi les fichiers, on note :

* **contributors.txt :** destiné à mentionner les auteurs ou contributeurs du package.
* **global.R** : Fichier de configuration globale. Il est exécuté une seule fois au lancement de l'application, avant ui.R et server.R. Il sert à charger des packages, lire des données, définir des fonctions communes, etc.
* **Server.R :** Contient la logique côté serveur : traitements, visualisations, lecture/écriture de données, calculs, etc. Il interagit avec l’UI à travers les entrées et sorties.
* **ui.R :** Définit l’interface utilisateur (UI) de l’application Shiny : menus, boutons, graphes, onglets, disposition de l’écran, etc.

1. **Man**

Ce dossier contient les fichiers de documentation des différentes fonctions du package, générées automatiquement par **roxygen2**. On y trouve les fichiers d’extensions **Rd** tels que :

1. **sdmApp.Rd :** Contient la documentation de la fonction **sdmApp()**
2. **sdmApp\_PA.Rd :** Contient la documentation de la fonction **sdmApp\_PA()**
3. **sdmApp\_RasterPlot.Rd :** Contient la documentation de la fonction **sdmApp\_ RasterPlot()**
4. **sdmApp\_TimesRasters.Rd :** Contient la documentation de la fonction **sdmApp\_TimesRasters()**
5. **sdmApp\_fold\_Exporer.Rd :** Contient la documentation de la fonction **sdmApp\_fold\_Explorer()**
6. **TESTS**

Ce dossier contient les tests unitaires et fonctionnels pour vérifier le bon fonctionnement du package. Il est composé d’unfichier **testthat.R** qui lance tous les tests du package et d’un sous dossier **Testthat** comprenant les tests unitaires, contenu dans les fichiers suivants.

* **test-sdmApp.Rd**
* **test-sdmApp\_PA.Rd**
* **test- sdmApp\_TimesRasters.Rd**
* **test-sdmApp\_fold\_Exporer.Rd**

Pour chaque fonction, c’est le bon fonctionnement de la multiplication qui est testée**.**

1. **Vignettes**

Ce dossier contient les documents explicatifs détaillés sur les fonctionnalités et l'utilisation du package. Il contient deux fichiers à savoir

* **.gitignore :** spécifie les fichiers à ne pas suivre par Git ; (.html et .R dans notre cas)
* **sdmApp.Rmd :** document expliquant le package sdmApp, ses auteurs, ses fonctionnalités, etc

1. **Fichiers**

Pour ce qui est des fichiers, nous pouvons citer :

* **.gitignore** : liste les fichiers et dossiers à ignorer par Git (fichiers temporaires, dossiers de build, etc.).
* **.Rbuildignore :** dans cette partie figurent les fichiers à ne pas inclure lors de la construction du package sdmApp.
* **\_pkgdown.yml** : fichier de configuration pour **pkgdown**, un outil qui génère un site web de documentation pour le package.
* **Codecov :** fichier de configuration pour **Codecov**, service mesurant la couverture de code par les tests.
* **cran-comments.md :** fichier contenant les commentaires pour les mainteneurs de CRAN lors de la soumission du package.
* **CRAN-RELEASE :** fichier généré lors de la soumission du package à CRAN, contenant des informations sur la version soumise.
* **DESCRIPTION :** fichier essentiel décrivant le package (nom, version, auteurs, dépendances, description, etc.).
* **NAMESPACE :** fichier définissant quelles fonctions du package sont exportées et quelles fonctions d'autres packages sont importées.
* **README :** fichier Markdown décrivant le package, son installation et son utilisation de base (ce que vous avez partagé précédemment).
* **sdmApp :** fichier projet RStudio **(.Rproj)** qui configure l'environnement de développement RStudio pour ce package.

1. **Différentes fonctionnalités du package**

Le package **sdmApp** renferme plusieurs fonctionnalités qui sont représentées par les différentes fonctions suivantes :

* **sdmApp()**

Elle est **l'interface utilisateur principale** du package. Elle charge dynamiquement les fichiers ui.R, server.R, global.R à partir du package ; **crée et lance l'application** avec l'interface utilisateur (ui) et la logique serveur (server). Une fois l’application lancée, plusieurs opérations peuvent être fait :

* Téléversement de fichiers raster et d'occurrence des espèces
* Analyse de corrélation entre couches raster
* Sélection des prédicteurs avec **CENFA**
* Validation croisée spatiale via **blockCV**
* Application de différents modèles de distribution d'espèces
* Export des résultats et du code R sous-jacent
* **sdmApp\_PA()**

Elle **affiche une carte** (via ggplot2) à partir d’un **objet raster binaire** représentant la **présence (1) en vert** ou **l’absence (0) en rouge** d’une espèce sur une zone géographique. En somme, elle prend en entré un raster binaire ayant pour chaque espèce l’information sur sa présence ou pas dans la zone et en se basant sur les données géographiques telles que la longitude et latitude, la carte est réalisée et est le résultat en sortie.

* **sdmApp\_TimesRasters()**

Elle sert à **masquer les absences** dans une carte de **probabilité d’occurrence** (par exemple issue d’un modèle), en **multipliant** cette carte par une **carte binaire** de présence/absence. Comme résultat, on obtient **uniquement les probabilités dans les zones de présence**. En somme, elle prend en entré une carte de probabilité d’occurrence et une carte binaire de présence/absence. En guise de sortie, nous avons un raster contenant uniquement les probabilités dans les zones de présences.

* **sdmApp\_fold\_Exporer()**

**Elle permet de** visualiser les folds (blocs de validation croisée spatiale) générés avec blockCV, en les superposant à une carte raster et aux points d’occurrence d’espèce. Affiche à la fois l’ensemble d’entrainement et l’ensemble de test.

Elle prend en entrée :

* Blocks : Objet SpatialBlock (de blockCV), contenant les folds.
* RasterLayer : Raster de fond (souvent des variables environnementales).
* SpeciesData : Données d’occurrence de l’espèce (format sf ou SpatialPoints).
* Num : Numéro du fold à explorer.

**Et retourne en sortie** deux cartes côte-à-côte, dont l’une est pour l’ensemble d’entraînement (training set)et l’autre pour l’ensemble de test (testing set).

**Elle permet ainsi donc, de v**érifier la cohérence de la distribution spatiale des folds, d’assurer que la séparation train/test est raisonnable et de visualiser comment les données de présence sont réparties.

* **sdmApp\_RasterPlot()**

Elle permet de créer une **jolie carte raster** en utilisant ggplot2, ce qui permet une **meilleure personnalisation** et une **intégration élégante** dans des Shiny apps