**LAPORAN**

**STUDI KASUS SUPERVISED - KLASIFIKASI**

****

**Kelompok**

**Nama Anggota :**

1. **Adie Gunawan Alwani (5200411486)**
2. **Alfia Candra Kusumapratiwi (5200411487)**
3. **Arieska Restu (5200411488)**

**S1 INFORMATIKA**

**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS TEKNOLOGI YOGYAKARTA**

**2021/2022**

Studi kasus 2a

Penjelasan Data

Dataset ini mencakup deskripsi sampel hipotetis yang sesuai dengan 23 spesies jamur insang di Agaricus dan Jamur Keluarga Lepiota yang diambil dari The Audubon Society Field Guide to North American Mushrooms (1981). Setiap spesies diidentifikasi sebagai pasti dapat dimakan (edible), pasti beracun (poisonous), atau tidak diketahui dapat dimakan dan tidak direkomendasikan.

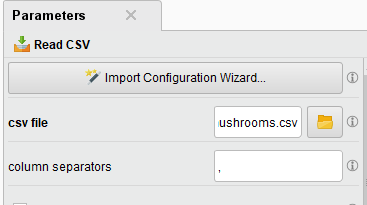
Algoritma Naïve Bayes

Langkah-langkah Algoritma Naïve Bayes

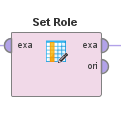
Langkah yang pertama yaitu kita menggunakan operator Read CSV untuk menambahkan datasetnya.



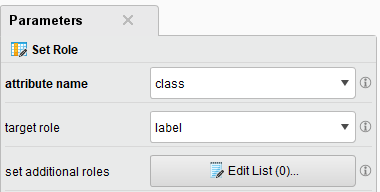
Cara menambahkan datasetnya yaitu dengan klik button yang bertuliskan **Import Configuration Wizard.** Maka otomatis akan diarahkan ke file manager, lalu pilih data mushroom yang sudah di download melalui kaggle.



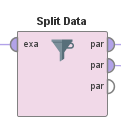
Kemudian kita perlu menambahkan operator **Set Role** yang berfungsi untuk membedakan baris penamaan atribut koordinat prediksi posisi yang akan dimasukkan kedalam kategori label.



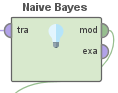
Setelah itu, pada parameters **Set Role** pada attribute namenya diubah menjadi kelas dan target rolenya menjadi label.



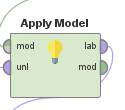
Langkah selanjutnya tambahkan operator **Split Data** yang berfungsi untuk membagi dataset menjadi partisi data latih dan data uji sesuai porsi yang ditentukan. Pada parameters **Split Data** tambahkan partisinya 0.8 dan 0.2.



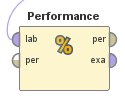
Kemudian drag and drop operator Naive Bayes yang berfungsi untuk memprediksi akurasinya.



Setelah itu tambahkan operator **Apply Model** yang digunakan untuk menerapkan model yang telah dilatih sebelumnya menggunakan data training pada data testing. Untuk operator ini kita tidak perlu mengubah pada bagian parameternya.

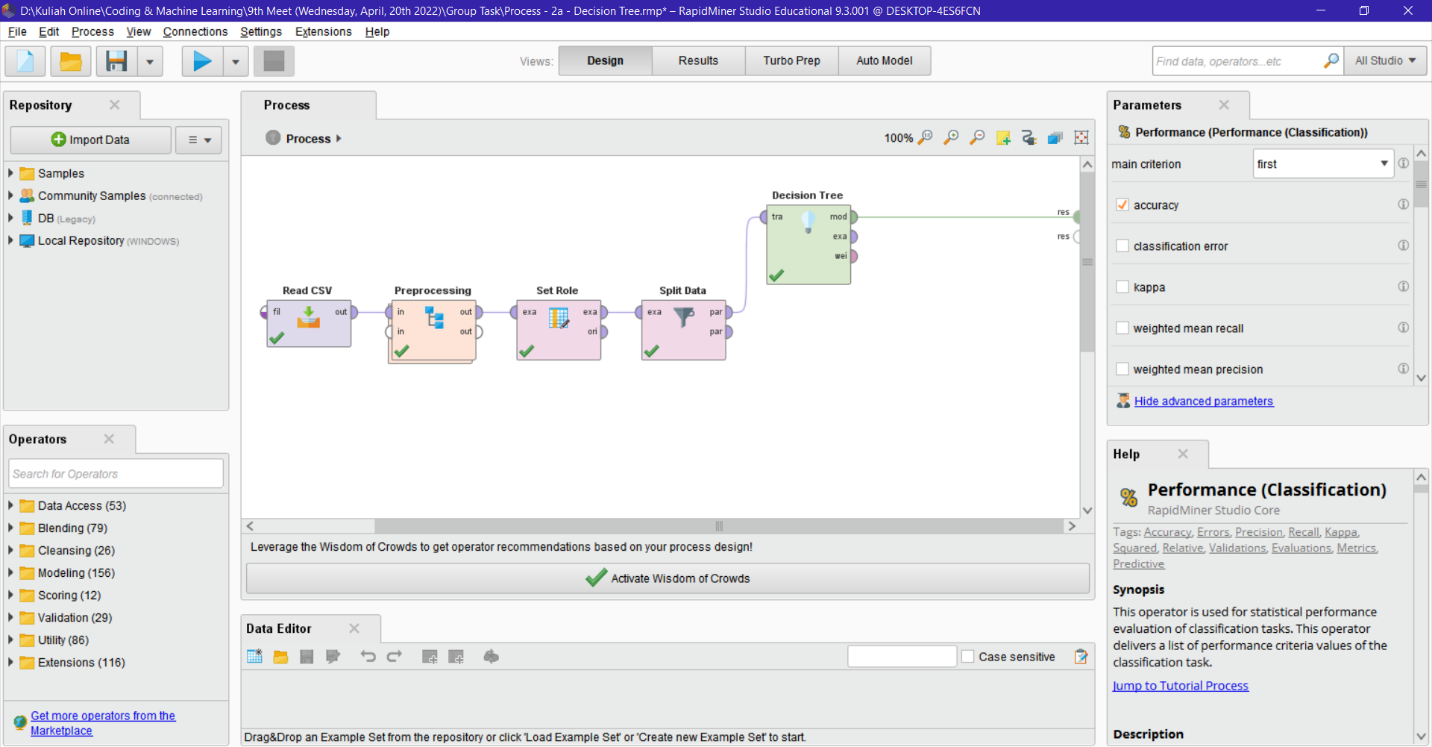


Langkah yang terakhir yaitu menambahkan operator **Performance(Classification)** yang berfungsi untuk mengevaluasi kinerja model yang memberikan daftar nilai kriteria kinerja secara otomatis sesuai dengan tugas yang diberikan.



Algoritma Decision Tree

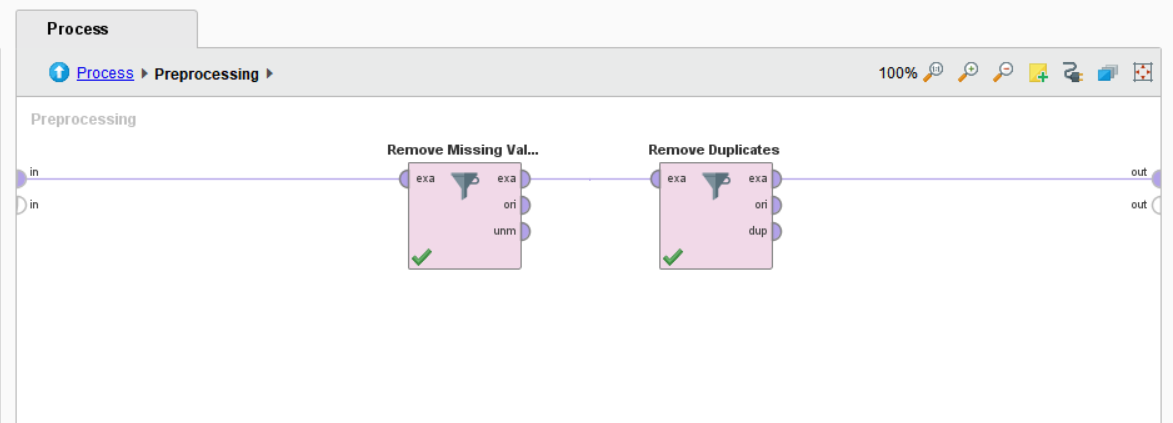
Langkah-langkah dalam Membangun Model



1. Read CSV

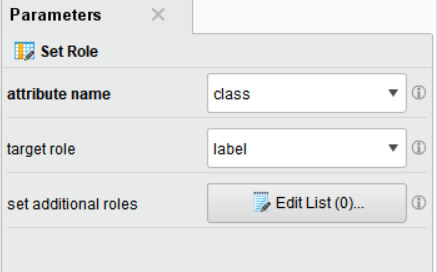
Proses diawali dengan membaca data dengan menggunakan operator Read CSV.

1. Preprocessing



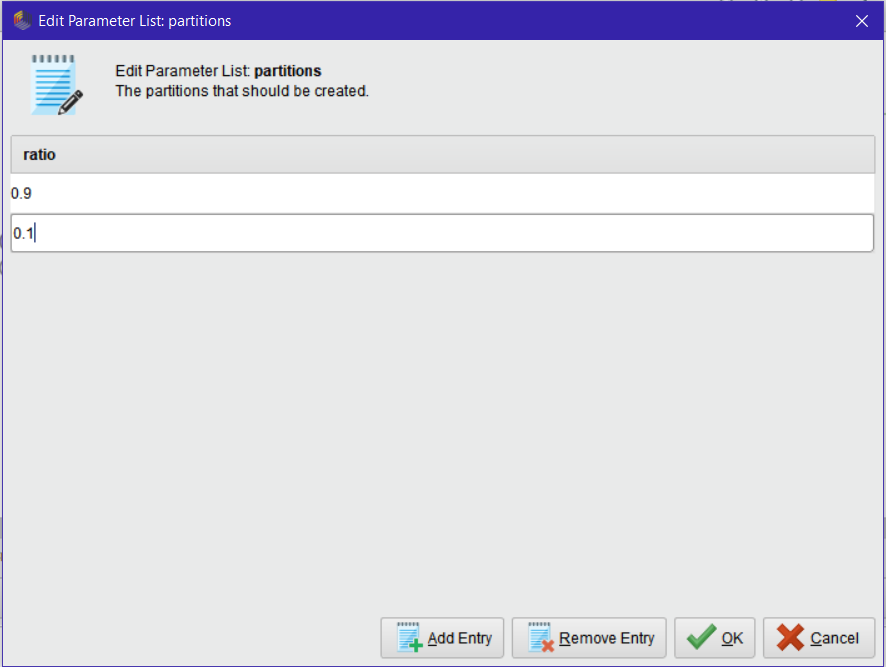
Setelah membaca datanya, kemudian dilanjutkan dengan melakukan preprocessing pada data. Pada preprocessing menghilangkan missing attribute dengan menggunakan operator **Filter Example**. Lalu dengan menggunakan operator **Remove Duplicates** akan menghilangkan data yang sama.

1. Set Role



Setelah melakukan preprocessing, kemudian memberi role label pada attribute class dengan menggunakan operator **Set Role**.

1. Split Data

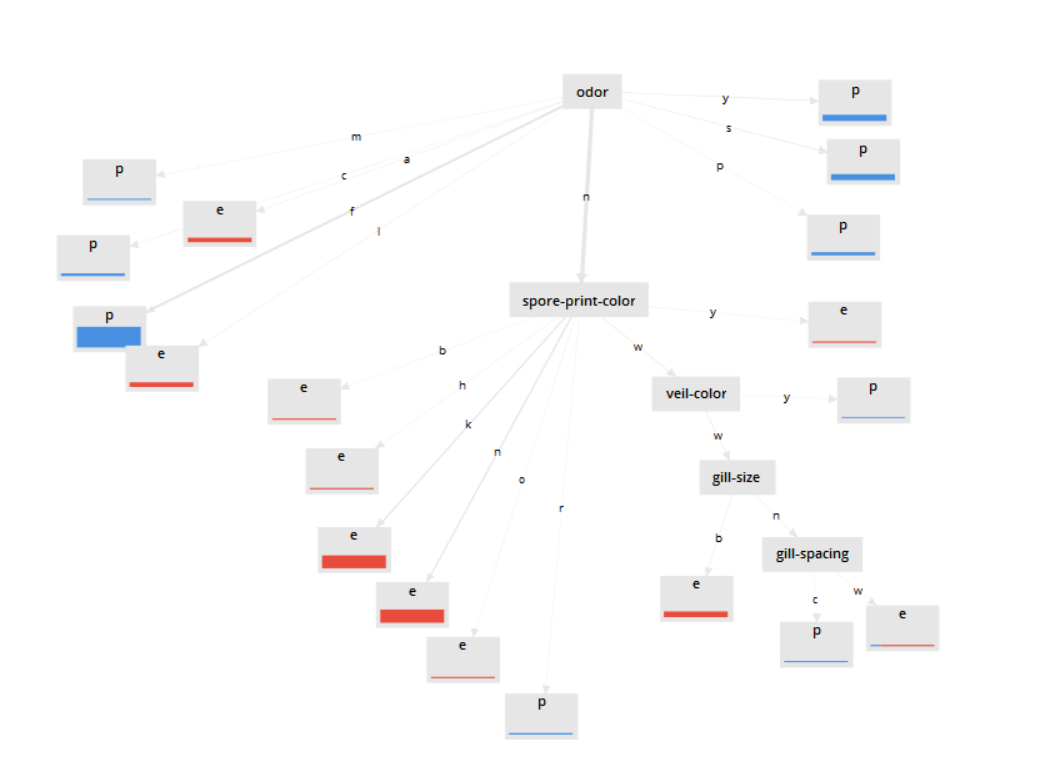


Selanjutnya dataset dipisah untuk dataset training dan testing dengan menggunakan operator **Split Data**. Dataset tersebut dipisah menjadi 9:1. Untuk 90% dari total datasetnya akan digunakan sebagai dataset training yang mana digunakan untuk melatih model. Sedangkan sisanya akan digunakan untuk melakukan testing dengan menggunakan model yang telah dilatih sebelumnya.

1. Decision Tree

Pada tahap ini akan melakukan proses training untuk melatih model dengan menggunakan 90% dari total datasetnya.

Penjelasan Model yang Didapatkan

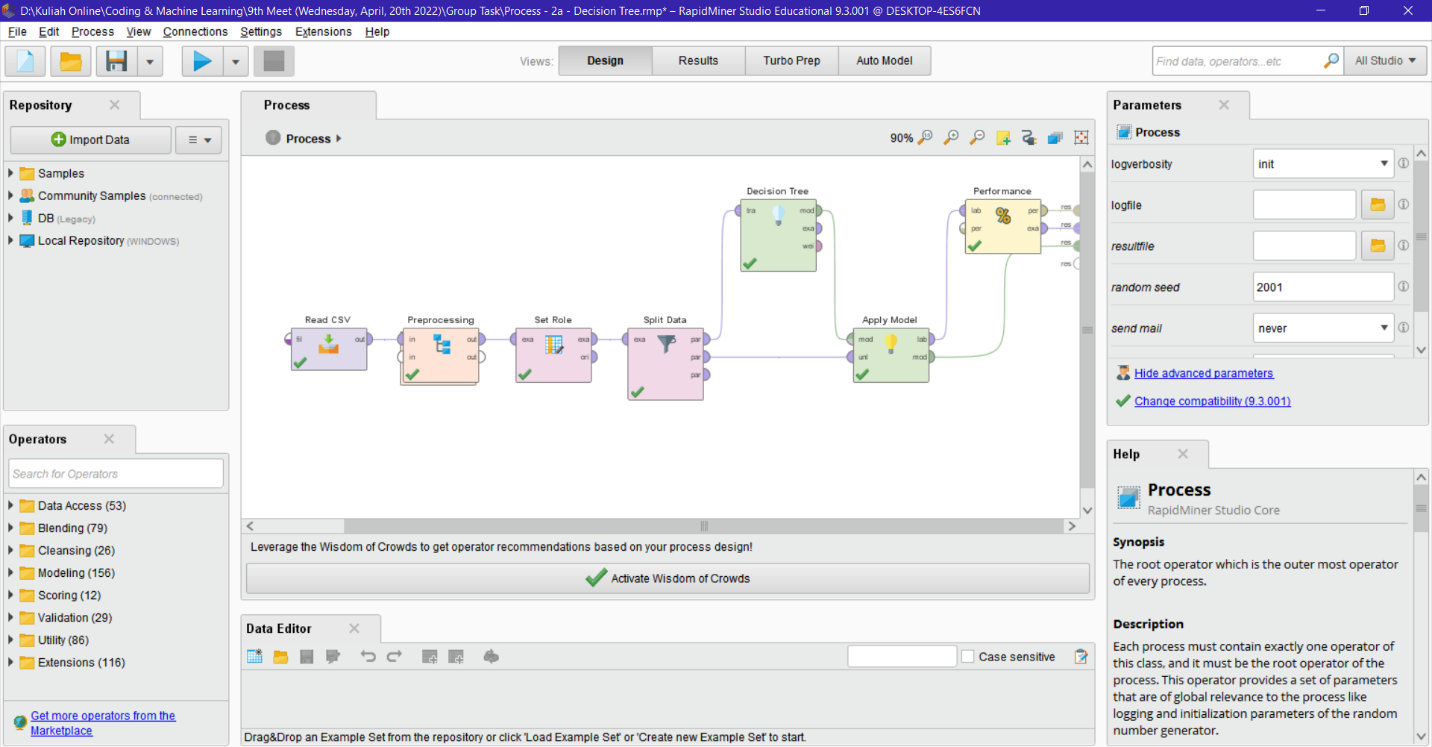


Dari model yang didapatkan, attribute yang menjadi root yaitu attribute **odor**. Apabila suatu data memiliki attribute **odor** yang bernilai **s** (spicy), **f** (foul), **y** (fishy), **p** (pungent), **c** (creosote), atau **m** (musty), maka data tersebut termasuk kelas **p** (poisonous). Sedangkan apabila dalam suatu data pada attribute **odor** memiliki nilai **a** (almond) atau **l** (anise), maka data tersebut termasuk kelas **e** (edible). Untuk data pada attribute **odor** yang memiliki nilai **n** (none), maka kelas dari data tersebut ditentukan berdasarkan nilai pada attribute **spore-print-color**.

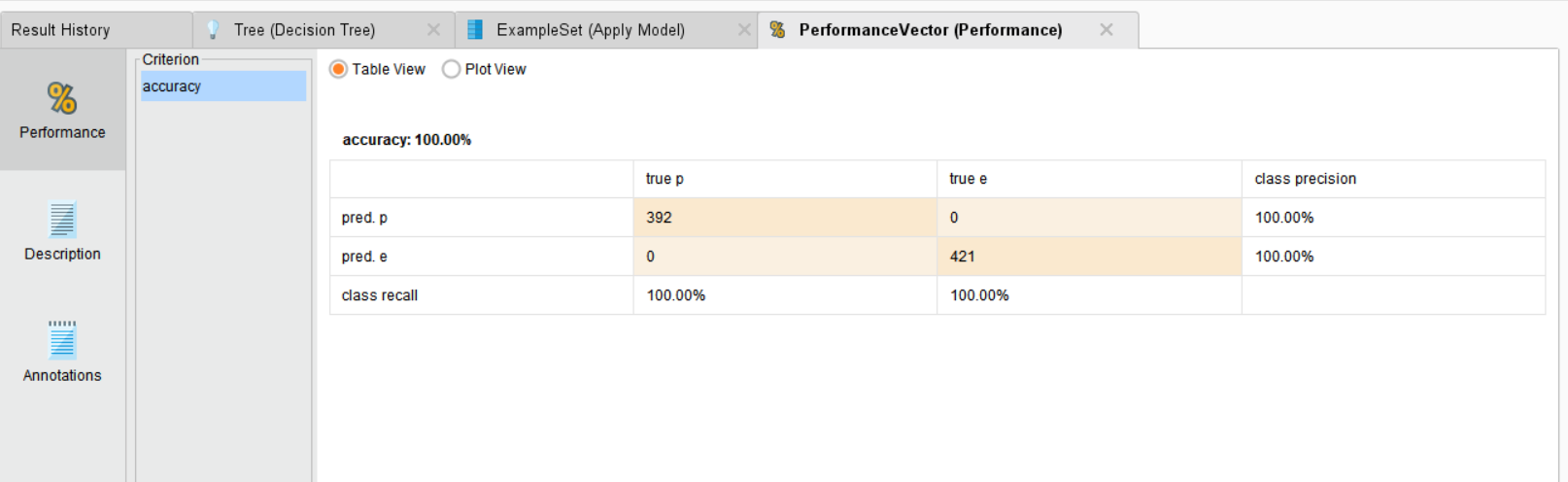
Apabila dalam suatu data pada attribute **odor** memiliki nilai **n** dan pada attribute **spore-print-color** memiliki nilai **k** (black), **b** (buff), **h** (chocolate), **n** (brown), **o** (orange), atau **y** (yellow), maka data tersebut termasuk kelas **e** (edible). Sedangkan apabila suatu data pada attribute **odor** memiliki nilai **n** dan pada attribute **spore-print-color** memiliki nilai **r** (green), maka data tersebut termasuk kelas **p** (poisonous). Namun, apabila suatu data pada attribute **odor** memiliki nilai **n** dan pada attribute **spore-print-color** memiliki nilai **w**, maka kelas dari data tersebut ditentukan berdasarkan nilai pada attribute **veil-color**.

Apabila suatu data pada attribute **odor** memiliki nilai **n**, pada attribute **spore-print-color** memiliki nilai **w**, dan pada attribute **veil-color** memiliki nilai **y** (yellow), maka data tersebut termasuk kelas p (poisonous). Sedangkan apabila pada **veil-color** nilainya **w** (white), maka kelas dari data tersebut ditentukan berdasarkan nilai pada attribute **gill-size**. Apabila suatu data pada attribute **odor** memiliki nilai **n**, pada attribute **spore-print-color** memiliki nilai **w**, pada attribute **veil-color** memiliki nilai **w**, dan pada attribute **gill-size** memiliki nilai **b** (broad), maka data tersebut termasuk kelas **e** (edible). Sedangkan apabila pada **gill-size** nilainya **n** (narrow), maka kelas dari data tersebut ditentukan berdasarkan nilai dari attribute **gill-spacing**. Apabila suatu data pada attribute **odor** memiliki nilai **n**, pada attribute **spore-print-color** memiliki nilai **w**, pada attribute **veil-color** memiliki nilai **w**, pada attribute **gill-size** memiliki nilai **n**, dan pada attribute **gill-spacing** memiliki nilai **c** (close), maka data tersebut termasuk kelas **p** (poisonous). Sedangkan apabila pada **gill-spacing** nilainya **w** (crowded), maka data tersebut termasuk kelas **e** (edible).

Mengklasifikasi Jamur



Setelah mendapatkan model, maka klasifikasi dari data jamur dapat dilakukan. Klasifikasi jamur dilakukan dengan menggunakan 10% dari total dataset dan dengan menggunakan operator **Apply Model**. Selanjutnya melakukan evaluasi terhadap kinerja statistik dari proses klasifikasi yang telah dilakukan. Evaluasi tersebut dilakukan dengan menggunakan operator **Performance (Classification)**. Evaluasi ini dilakukan untuk melihat seberapa besar akurasi yang didapat. Akurasi dari klasifikasi jamur yang telah dilakukan yaitu sebagai berikut.



Studi kasus 2b

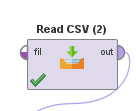
Penjelasan Data

Permasalahan pada data yang kami dapatkan adalah memprediksi keberhasilan dari suatu startup. Startup atau start-up adalah perusahaan atau proyek yang dimulai oleh seorang wirausahawan untuk mencari, mengembangkan, dan memvalidasi model ekonomi yang terukur. Data yang kami dapatkan bersumber dari kaggle.com, lebih tepatnya <https://www.kaggle.com/datasets/manishkc06/startup-success-prediction?resource=download>.

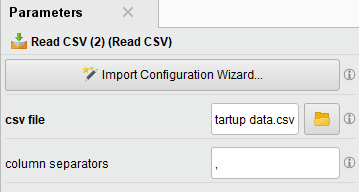
Algoritma Naïve Bayes

Langkah-langkah Algoritma Naïve Bayes

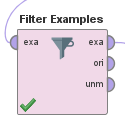
Langkah pertama yaitu tambahkan operator **Read CSV** untuk menambahkan dataset.



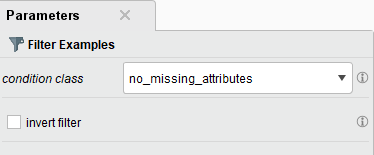
Cara menambahkan datasetnya yaitu dengan klik button yang bertuliskan **Import Configuration Wizard.** Maka otomatis akan diarahkan ke file manager, lalu pilih data mushroom yang sudah di download melalui kaggle.



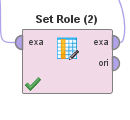
Kemudian tambahkan operator **Filter Examples** untuk menghilangkan kolom yang tidak bernilai atau kosong.



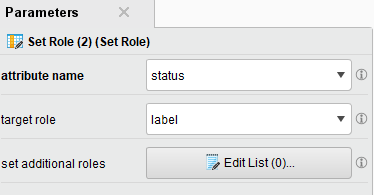
Selanjutnya pada parameters **Filter Example** ubah menjadi **no\_missing\_attribute** pada condition class.



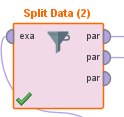
Kemudian kita perlu menambahkan operator **Set Role** yang berfungsi untuk membedakan baris penamaan atribut koordinat prediksi posisi yang akan dimasukkan kedalam kategori label.



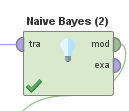
Setelah itu, pada parameters **Set Role** pada attribute namenya diubah menjadi kelas dan target rolenya menjadi label.



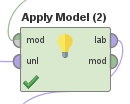
Langkah selanjutnya tambahkan operator **Split Data** yang berfungsi untuk membagi dataset menjadi partisi data latih dan data uji sesuai porsi yang ditentukan. Pada parameters **Split Data** tambahkan partisinya 0.8 dan 0.2.



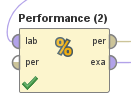
Kemudian drag and drop operator Naive Bayes yang berfungsi untuk memprediksi akurasinya.



Setelah itu tambahkan operator **Apply Model** yang digunakan untuk menerapkan model yang telah dilatih sebelumnya menggunakan data training pada data testing. Untuk operator ini kita tidak perlu mengubah pada bagian parameternya.

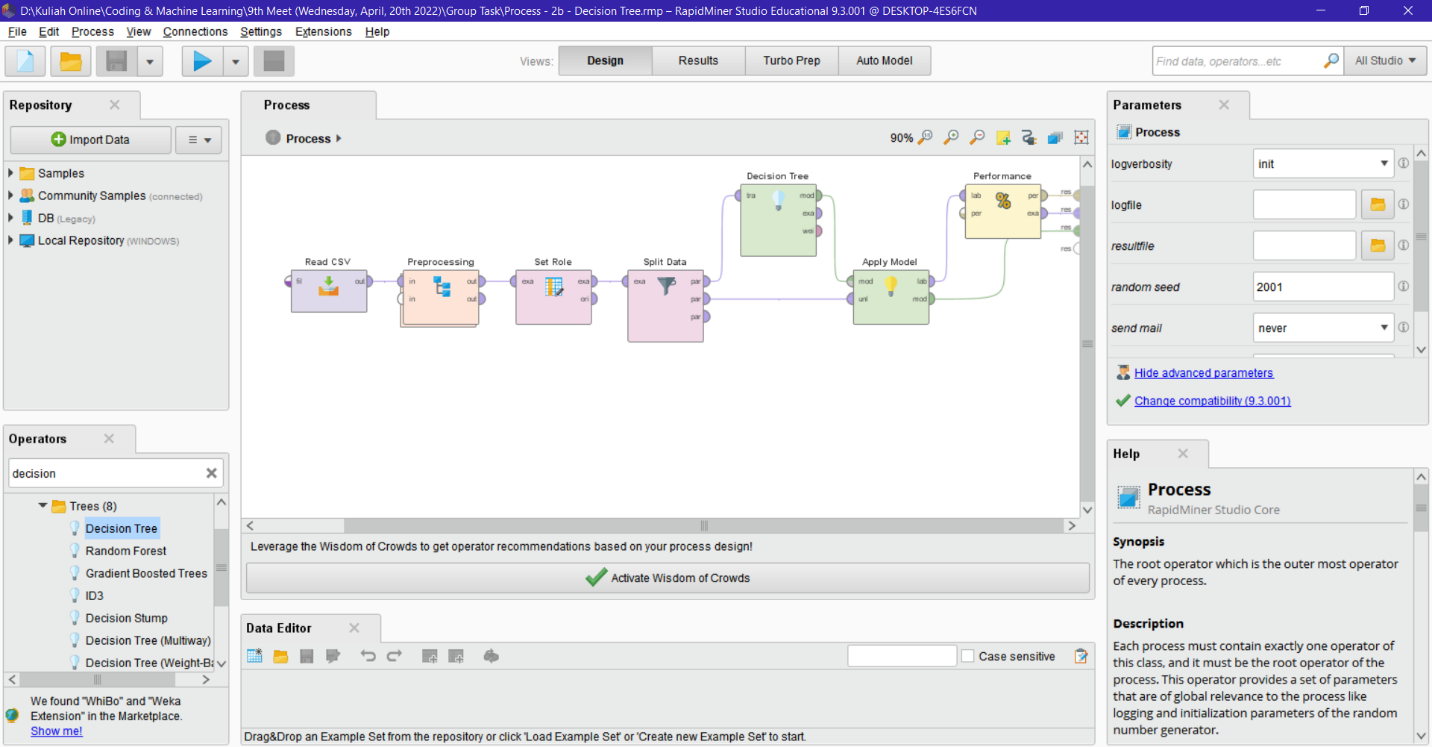


Langkah yang terakhir yaitu menambahkan operator **Performance(Classification)** yang berfungsi untuk mengevaluasi kinerja model yang memberikan daftar nilai kriteria kinerja secara otomatis sesuai dengan tugas yang diberikan.



Algoritma Decision Tree

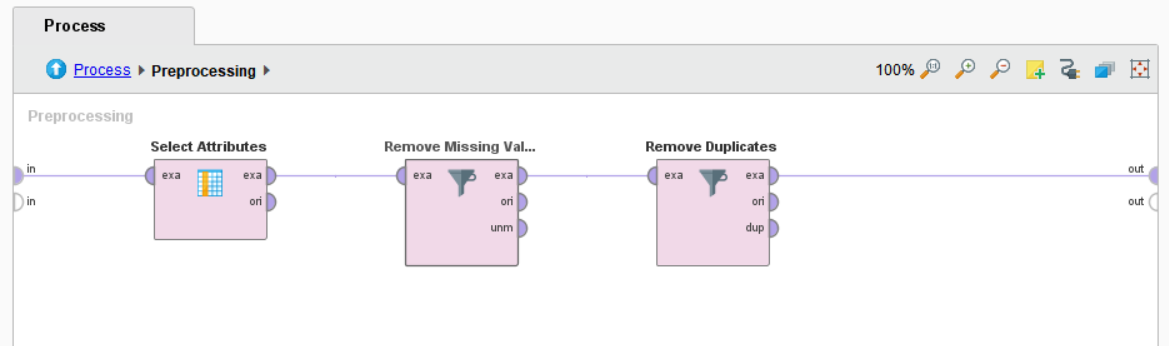
Langkah-langkah dalam Mengembangkan Model



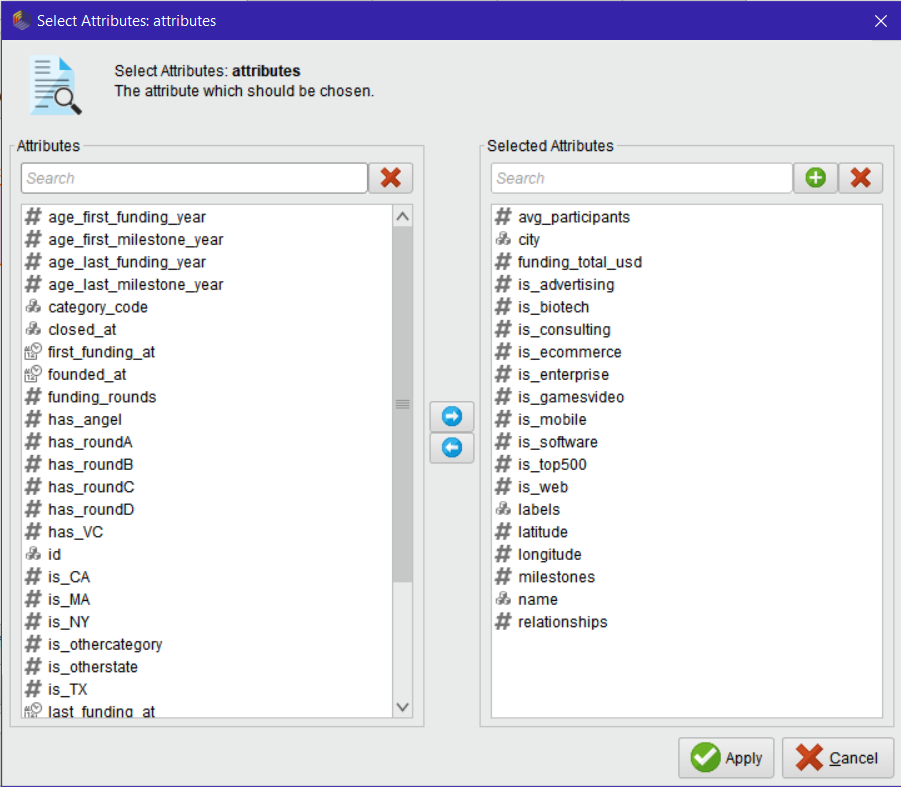
1. Read CSV

Proses diawali dengan membaca data dengan menggunakan operator **Read CSV**.

1. Preprocessing



Setelah membaca datanya, kemudian dilanjutkan dengan melakukan preprocessing pada data. Pada preprocessing, dilakukan pemilihan attribute dari datanya dengan menggunakan operator **Select Attrribute**. Attribute yang akan digunakan yaitu sebagai berikut.



Selanjutnya yaitu menghilangkan missing attribute dengan menggunakan operator **Filter Example**. Lalu dengan menggunakan operator **Remove Duplicates** akan menghilangkan data yang sama.

1. Set Role

Setelah melakukan preprocessing, kemudian memberi role label pada attribute labels dengan menggunakan operator **Set Role**.

1. Split Data

Selanjutnya dataset dipisah untuk dataset training dan testing dengan menggunakan operator **Split Data**. Dataset tersebut dipisah menjadi 9:1. Untuk 90% dari total datasetnya akan digunakan sebagai dataset training yang mana digunakan untuk melatih model. Sedangkan sisanya akan digunakan untuk melakukan testing dengan menggunakan model yang telah dilatih sebelumnya.

1. Decision Tree

Pada tahap ini akan melakukan proses training untuk melatih model dengan menggunakan 90% dari total datasetnya.

1. Apply Model

Setelah model dilatih, maka melakukan testing untuk 10% dari total dataset dengan menggunakan operator **Apply Model**.

1. Performance

Selanjutnya yaitu melakukan evaluasi terhadap kinerja statistik dari proses klasifikasi yang telah dilakukan. Evaluasi tersebut dilakukan dengan menggunakan operator **Performance (Classification)**. Evaluasi ini dilakukan untuk melihat seberapa besar akurasi yang didapat. Akurasi dari proses yang telah dilakukan yaitu sebagai berikut.

