El futuro digital es de todos

MinTIC

Blockchain & analítica de datos

para industrias digitales

etraining
Diseñamos nuevas formas de aprender

tecnalia Colombia
Inspiring Business

El futuro digital es de todos

MinTIC

Analítica de Datos

Sesión N°14

etraining
Diseñamos nuevas formas de aprender

tecnalia Colombia
Inspiring Business

Ejercicio 1

PyCaret es una herramienta de Inteligencia Artificial low code y Open Source. Recordemos que las herramientas low code facilitan la implementación de los algoritmos usando muy poco código fuente. En este ejercicio vamos a probar un conjunto de algoritmos, para hacer una selección:

En Google Colab debemos instalar el paquete:

```
!pip install pycaret
```

Sin embargo, vamos a comprobar la versión con la siguiente línea:

```
from pycaret.utils import version
```

Ahora podemos cargar y observar todos los repositorios disponibles para pruebas:

```
from pycaret.datasets import get_data
index = get_data('index')
```

Cargamos el conjunto de datos ‘juice’, la información del dataset se puede encontrar en https://rdrr.io/cran/ISLR/man/OJ.html:

```
data = get_data('juice')
```

Configuramos el entrenamiento para clasificar, el objetivo es la variable 'Purchase', que representa Citrus Hill o Minute Maid Orange Juice:

```
from pycaret.classification import *
clf1 = setup(data, target = 'Purchase', session_id=123, log_experiment=True, experiment_name='juice1')
```

Ejecutamos el entrenamiento comparativo de los modelos:

```
best_model = compare_models()
```

El futuro digital es de todos | MinTIC

Ejercicio 1

La salida debe verse así:

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
ridge	Ridge Classifier	0.8341	0.0000	0.7594	0.8024	0.7780	0.6460	0.6491	0.014
lr	Logistic Regression	0.8328	0.8922	0.7382	0.8105	0.7711	0.6400	0.6432	0.359
lda	Linear Discriminant Analysis	0.8328	0.8921	0.7664	0.7951	0.7782	0.6443	0.6470	0.017
gbc	Gradient Boosting Classifier	0.8155	0.8873	0.7555	0.7648	0.7581	0.6091	0.6113	0.129
ada	Ada Boost Classifier	0.8154	0.8847	0.7278	0.7808	0.7517	0.6052	0.6078	0.111
lightgbm	Light Gradient Boosting Machine	0.7941	0.8758	0.7278	0.7353	0.7308	0.5641	0.5649	0.094
rf	Random Forest Classifier	0.7887	0.8687	0.7171	0.7287	0.7204	0.5509	0.5534	0.515
et	Extra Trees Classifier	0.7713	0.8357	0.7032	0.7028	0.7002	0.5159	0.5184	0.463
nb	Naive Bayes	0.7700	0.8420	0.7836	0.6750	0.7229	0.5285	0.5357	0.017
dt	Decision Tree Classifier	0.7473	0.7401	0.6683	0.6726	0.6687	0.4646	0.4663	0.018
qda	Quadratic Discriminant Analysis	0.7419	0.8408	0.8225	0.6295	0.7096	0.4858	0.5048	0.018
knn	K Neighbors Classifier	0.7367	0.7695	0.6166	0.6788	0.6414	0.4348	0.4400	0.118
svm	SVM - Linear Kernel	0.5417	0.0000	0.3966	0.1621	0.2295	0.0245	0.0347	0.017

Ejercicio 1

Si necesitamos un modelo en particular, podemos cargar el listado con:

ID	Name	Reference	Turbo
lr	Logistic Regression	sklearn.linear_model._logistic.LogisticRegression	True
knn	K Neighbors Classifier	sklearn.neighbors._classification.KNeighborsCl...	True
nb	Naive Bayes	sklearn.naive_bayes.GaussianNB	True
dt	Decision Tree Classifier	sklearn.tree._classes.DecisionTreeClassifier	True
svm	SVM - Linear Kernel	sklearn.linear_model._stochastic_gradient.SGDC...	True
rbfsvm	SVM - Radial Kernel	sklearn.svm._classes.SVC	False
gpc	Gaussian Process Classifier	sklearn.gaussian_process._gpc.GaussianProcessC...	False
mlp	MLP Classifier	sklearn.neural_network._multilayer_perceptron....	False
ridge	Ridge Classifier	sklearn.linear_model._ridge.RidgeClassifier	True
rf	Random Forest Classifier	sklearn.ensemble._forest.RandomForestClassifier	True
qda	Quadratic Discriminant Analysis	sklearn.discriminant_analysis.QuadraticDiscrim...	True
ada	Ada Boost Classifier	sklearn.ensemble._weight_boosting.AdaBoostClas...	True
gbc	Gradient Boosting Classifier	sklearn.ensemble._gb.GradientBoostingClassifier	True
lda	Linear Discriminant Analysis	sklearn.discriminant_analysis.LinearDiscrimina...	True
et	Extra Trees Classifier	sklearn.ensemble._forest.ExtraTreesClassifier	True
lightgbm	Light Gradient Boosting Machine	lightgbm.sklearn.LGBMClassifier	True

De la columna id podemos tomar el nombre con el que lo interpreta la herramienta, si necesitamos usar por ejemplo un Multi Layer Perceptron, la Sigla es MLP:

mlp = create_model('mlp')

Ejercicio 2

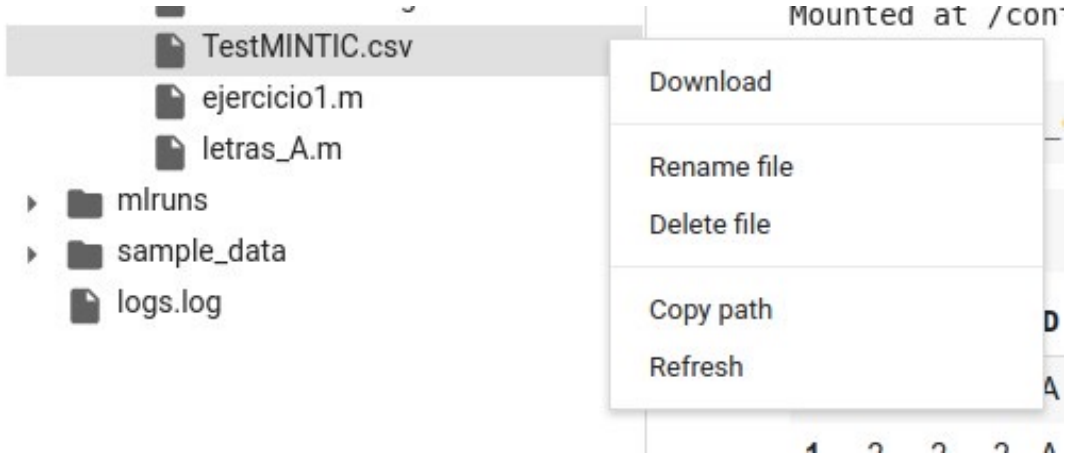
Ahora vamos a cargar nuestros propios datos, cargamos el archivo en formato csv a nuestro Google Drive, luego ejecutamos la siguiente línea:

```
from google.colab import drive
drive.mount('/content/drive')
```

Ahora cargamos el siguiente paquete:

```
import pandas as pd
```

Buscamos en la ventana derecha la ruta del archivo “Copy path”:



Y cargamos el archivo con la ruta:

```
data = pd.read_csv('/content/drive/MyDrive/TestMINTIC.csv')
```

El futuro digital es de todos

MinTIC

Gracias

etraining
Diseñamos nuevas formas de aprender

tecnalia Colombia
Inspiring Business