from sklearn import datasets

from sklearn.model\_selection import train\_test\_split

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score

#from sklearn.externals import joblib

import numpy as np

import joblib

def train(X,y):

    # train test split

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3)

    knn = KNeighborsClassifier(n\_neighbors=1)

    # fit the model

    knn.fit(X\_train, y\_train)

    preds = knn.predict(X\_test)

    acc = accuracy\_score(y\_test, preds)

    print(f'Successfully trained model accuracy of {acc:.2f}')

    return knn

iris\_data = datasets.load\_iris()

X = iris\_data['data']

y = iris\_data['target']

labels = {0 : 'iris-setosa',

          1 : 'iris-versicolor',

          2 : 'iris-virginica'}

# rename integer labels to actual flower names

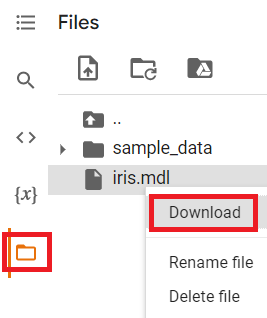
y = np.vectorize(labels.\_\_getitem\_\_)(y)

modelo\_knn = train(X,y)

# serialize model

joblib.dump(modelo\_knn, 'iris.mdl')

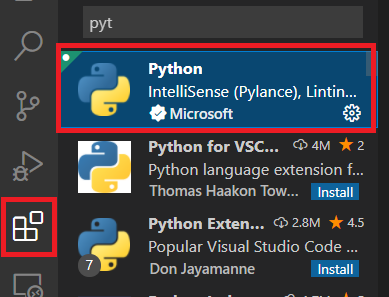
Download modelo



VS code

Open Folder

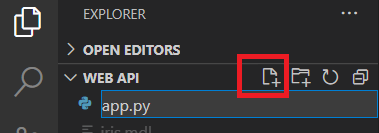
If it is not installed



Verify Python version

python3 --version

New file app.py



from flask import Flask

from flask\_restful import Api, Resource, reqparse

from sklearn.externals import joblib

import numpy as np

APP = Flask(\_\_name\_\_)

API = Api(APP)

#modelo previamente salvado

IRIS\_MODEL = joblib.load('iris.mdl')

class Predict(Resource):

@staticmethod

def post():

parser = reqparse.RequestParser()

parser.add\_argument('petal\_length')

parser.add\_argument('petal\_width')

parser.add\_argument('sepal\_length')

parser.add\_argument('sepal\_width')

args = parser.parse\_args() # creates dict

X\_new = np.fromiter(args.values(), dtype=float)

# convert input to array

resultado = {'Prediction': IRIS\_MODEL.predict([X\_new])[0]}

return resultado, 200

API.add\_resource(Predict, '/predict')

if \_\_name\_\_ == '\_\_main\_\_':

APP.run(debug=True, port='1080')

@app.route(‘/predict’, methods=[‘POST’,’GET’])

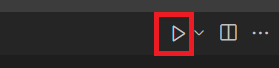
def predict\_lofic():

query = request.args

campo1 = float(query.get(‘campo1’))

resultado = modelo.predict([adf, adfd, adfd

return jsonify({‘prediction’: resultado})



PS C:\Proyectos\ML\Web API> & python "c:/Proyectos/ML/Web API/app.py"

Python was not found; run without arguments to install from the Microsoft Store, or disable this shortcut from Settings > Manage App Execution Aliases.

PS C:\Proyectos\ML\Web API>

Install Python

import requests

url = 'http://127.0.0.1:1080/predict'

# localhost + port / endpoint

body = {

"petal\_length": 2,

"sepal\_length": 2,

"petal\_width": 0.5,

"sepal\_width": 3

}

response = requests.post(url, data=body)

response.json()