Double-click (or enter) to edit

Import library

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Importing the dataset

```
dataset = pd.read_csv('data.csv')
X = dataset.iloc[:, 1:-1].values
y = dataset.iloc[:, -1].values
```

```
print(X)
print(y)
```

```
[[2141.       2154.       2121.350098 2131.550049 2124.715088]
 [2120.       2123.899902 2088.       2097.050049 2090.325684]
 [2118.       2122.       2077.       2081.850098 2075.174316]
 ...
 [2149.350098 2203.       2128.149902 2173.5      2173.5     ]
 [2168.850098 2185.199951 2147.850098 2164.25     2164.25    ]
 [2174.       2186.800049 2152.600098 2172.649902 2172.649902]]
[15731396 10401212 11667129 15098991  8947563 27630190 12961252 12537520
 28648125 17821397 12437162  8577634 13141644 10119064 12643749 25525961
 64750460 27445768 20335273 12542824 15668979 11919927 15264068 15519031
 16056620 19839282 13764841 13095028  9076248 11926328 11290667  9591456
  8732869  8497216 24538346  9727405  8558745  6565515 10666190 12341434
  9246702  9960828 14398973  8529573 15728986 14215215 10809307 17225212
 16835005 13809929 14147418 15700946 45857806 40931170 37003111 17170274
 30770080 16539467 17045147 26178477 18481466 20946864 21479385 14030652
 12828008 26522972 20918665 14277083 15062376 11924527 21845931  9114939
 10240168 12822945  8521388  8418767 20030506 13464375  7414229 12434745
  7989830  8561406  8565904  9346460  8522215 20368545 12986606  8999898
  9588577  7947719  8589407 10173132  8667516  4622002 11312992 11132803
 21414270 14918406 12709792 15371556 18996047 12284876  9946818  9503790
 16198856 14771048 14271669 18038987 14090818 25016570 19553809 15722291
 20173258 15774504 19138414 13984228 11826848  9344470  9776136  9047308
  9799560 19631870 14074374  7287590  9886093 10957388  8605531 10800704
 10985697 11834752  4987735 16085897 17297575  8159670  7915073 14733134
  9892597 11773630  9002404  6993792  5316182  7783173  7706170  6402757
  8865521  9528809 19284892  8571196  8039865  7763726 10153757  9313160
  9433842  7499740  5410307  6864856  6465241 11198918  7092878  6478482
  9646031  8958261  9102492  7225679  8527967  7939490  6687573  5459016
  9620785  9226547  7902002  8035915  9150974 10909942 10083693  5719649
  6749281  5671163  6433879  6220217  6081627  6134079  5479424  5567958
  7530294  5271497  4977555  3946636  6821015  3452113 12377100 26060864
 27285782 12928379 11366816 11064116  6677278  7539326  4464889  5265326
  6532832  6351520  9383416  7002898  4982975  7266765 13553801  5624044
  5940386  7685796 42209687 25546334  9357852 12526981  9120556  5784627
  6500973  5400604  4647361  4824931  5162088  4170886  3962111  3099956
  3937768  4360415  4074206  3855577  3111185  3679260  4585938  5717830
  4995516  5952355  6352886  5238304  5041934  5453682  6077861  9807831
  7670583  3458546  5500708  4238859  3755507  5898384 10123204  5841743
  4650008]
```

```
y = y.reshape(len(y), 1)
y
```

```
       [13553801],
       [ 5624044],
       [ 5940386],
       [ 7685796],
       [42209687],
       [25546334],
       [ 9357852],
       [12526981],
       [ 9120556],
       [ 5784627],
       [ 6500973],
       [ 5400604],
       [ 4647361],
       [ 4824931],
       [ 5162088],
       [ 4170886],
       [ 3962111],
       [ 3099956],
       [ 3937768],
       [ 4360415],
       [ 4074206],
       [ 3855577],
       [ 3111185],
       [ 3679260],
       [ 4585938],
       [ 5717830],
       [ 4995516],
       [ 5952355],
       [ 6352886],
       [ 5238304],
       [ 5041934],
       [ 5453682],
       [ 6077861],
       [ 9807831],
       [ 7670583],
       [ 3458546],
       [ 5500708],
       [ 4238859],
       [ 3755507],
       [ 5898384],
       [10123204],
       [ 5841743],
       [ 4650008]])
```

Splitting Dataset into Train and Test

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc_x = StandardScaler()
sc_y = StandardScaler()
X_train = sc_x.fit_transform(X_train)
y_train = sc_y.fit_transform(y_train)
```

Training the SVR model on the training set

```
from sklearn.svm import SVR
regressor = SVR(kernel = 'rbf')
regressor.fit(X_train, y_train)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d a
  y = column_or_1d(y, warn=True)
▾ SVR
SVR()
```

Predicting the test set results

```
y_pred = sc_y.inverse_transform(regressor.predict(sc_x.transform(X_test)))
np.set_printoptions(precision=2)
print(np.concatenate((y_pred.reshape(len(y_pred), 1), y_test.rehape(len(y_test), 1))))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-44-036c16dd5562> in <cell line: 1>()
----> 1 y_pred = sc_y.inverse_transform(regressor.predict(sc_x.transform(X_test)))
      2 np.set_printoptions(precision=2)
      3 print(np.concatenate((y_pred.reshape(len(y_pred), 1), y_test.rehape(len(y_test), 1))))

                                 1 frames
/usr/local/lib/python3.9/dist-packages/sklearn/utils/validation.py in check_array(array, accept_sparse, accept_large_sparse, dtype,
order, copy, force_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator, input_name)
    900             # If input is 1D raise error
    901             if array.ndim == 1:
--> 902                 raise ValueError(
    903                     "Expected 2D array, got 1D array instead:\narray={}.\n"
    904                     "Reshape your data either using array.reshape(-1, 1) if "

ValueError: Expected 2D array, got 1D array instead:
array=[-0.51402477 -0.06790167 -0.00129973 -0.05659866  0.03922851 -0.44306128
 -0.55020851 -0.4510927  -0.66199156 -0.6221312   0.14607555 -0.2149928
 -0.42165366 -0.16752322 -0.25360715 -0.03070679 -0.31591791 -0.09792665
 -0.62135195 -0.38037292 -0.11227181  0.44180026 -0.10363613 -0.48934615
 -0.16648762 -0.12636204 -0.24522663  0.44042338 -0.22332495  0.43651636
 -0.22177642 -0.01375416 -0.52470598 -0.2882943  -0.39687367  0.3641901
  0.31415166 -0.27157955 -0.62412496 -0.29553957 -0.44264198 -0.49196125
 -0.33934788 -0.34253064 -0.37499297 -0.22746391 -0.34642627 -0.25042194
 -0.32143222 -0.66591123].
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single
sample.
```

SEARCH STACK OVERFLOW

Evaluating the model performance

```
from sklearn.metrics import r2_score
r2_score(y_test, y_pred)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-45-7fcfd38683ff> in <cell line: 2>()
      1 from sklearn.metrics import r2_score
----> 2 r2_score(y_test, y_pred)

NameError: name 'y_pred' is not defined
```

SEARCH STACK OVERFLOW

Colab paid products - Cancel contracts here

✓ 0s completed at 6:46 AM