Dimensionality Reduction Using Feature Selection

```
from sklearn import datasets
from sklearn.feature_selection import VarianceThreshold

iris = datasets.load_iris()

X = iris.data
y = iris.target

thresholder = VarianceThreshold(threshold=.5)

X_high_variance = thresholder. fit_transform(X)

X_high_variance[0:5]
array = ([[5.1, 1.4, 0.2],
[4.9, 1.4, 0.2],
[4.7, 1.3, 0.2],
[4.6, 1.5, 0.2],
[5. , 1.4, 0.211]])

thresholder.fit(X).variances_
```

```
array([0.68112222, 0.18871289, 3.09550267, 0.57713289])
```

```
import numpy as np
import pandas as pd

# Load libraries import pandas as pd import numpy as np
# Create feature matrix with two highly correlated features
X = np.array([
    [1, 1, 1],
     [2, 2, 0],
  [3, 3, 1],
   [4, 4, 0],
    [5, 5, 1],
     [6, 6, 0],
      [7, 7, 1],
       [8, 7, 0],
        [9, 7, 1]])
# Convert feature matrix into DataFrame
df = pd.DataFrame(X)
#View the data frame
df
```

	0	1	2
0	1	1	1
1	2	2	0
2	3	3	1
3	4	4	0
4	5	5	1
5	6	6	0
6	7	7	1
7	8	7	0
8	9	7	1

```
corr_matrix = df.corr().abs()

upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(bool))

to_drop = [column for column in upper.columns if any (upper [column] > 0.95)]
df.drop(df[to_drop], axis=1)
```

	0	2
0	1	1
1	2	0
2	3	1
3	4	0
4	5	1
5	6	0
6	7	1
7	8	0
8	9	1

```
df.corr()
```

	0	1	2
0	1.000000	0.976103	0.000000
1	0.976103	1.000000	-0.034503
2	0.000000	-0.034503	1.000000

```
upper
```

	0	1	2
0	NaN	0.976103	0.000000
1	NaN	NaN	0.034503
2	NaN	NaN	NaN

```
df.drop(df[to_drop], axis =1)
```

	0	2
0	1	1
1	2	0
2	3	1
3	4	0
4	5	1
5	6	0
6	7	1
7	8	0
8	9	1

```
# Load libraries
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
#Load iris data
iris = load_iris()
# Create features and target
X = iris.data
y= iris.target
# Convert to categorical data by converting data to integers
X = X.astype (int)
# Select two features with highest chi-squared statistics
chi2_selector= SelectKBest (chi2, k=2)
X_kbest = chi2_selector.fit_transform(X, y)
# Show results
print('Original number of features:', X.shape[1])
print('Reduced number of features:', X_kbest.shape[1])
```

```
Original number of features: 4
Reduced number of features: 2
```

```
# Load libraries
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
# Load iris data
iris = load_iris()
# Create features and target
X = iris.data
y= iris.target
# Create an SelectKBest object to select features with two best ANOVA F-Values
fvalue_selector = SelectKBest (f_classif, k=2)
# Apply the SelectKBest object to the features and target
X_kbest = fvalue_selector.fit_transform(X, y)
# Show results
print('Original number of features: ', X.shape[1])
print('Reduced number of features: ', X_kbest.shape[1])
```

```
Original number of features:  4
Reduced number of features:  2
```

Start coding or generate with AI.

Double-click (or enter) to edit