Write a Python program to demonstrate the creation and loading of different types of datasets using pandas and scikit-learn, compute mean, median, mode, variance, and standard deviation, and perform data preprocessing techniques including reshaping, filtering, merging, handling missing values, and min-max normalization.

Create datasets using pandas

Generate sample datasets using pandas DataFrames.

```
import pandas as pd

data1 = {
    'ID': [1, 2, 3, 4, 5],
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
    'Age': [25, 30, 35, 28, 32],
    'Salary': [50000, 60000, 75000, 55000, 70000]
}

df1 = pd.DataFrame(data1)

data2 = {
    'ID': [1, 2, 3, 6, 7],
    'Department': ['Sales', 'IT', 'Marketing', 'Sales', 'IT'],
    'Location': ['New York', 'San Francisco', 'Los Angeles', 'New York', 'San Francisco']
}

df2 = pd.DataFrame(data2)

display(df1)
display(df2)
```

	ID	Name	Age	Salary
0	1	Alice	25	50000
1	2	Bob	30	60000
2	3	Charlie	35	75000
3	4	David	28	55000
4	5	Eve	32	70000

	ID	Department	Location
0	1	Sales	New York
1	2	IT	San Francisco
2	3	Marketing	Los Angeles
3	6	Sales	New York
4	7	IT	San Francisco

Load datasets using pandas

Demonstrate how to load data from CSV files into pandas DataFrames.

```
import io

csv_data = """ID,Product,Price
101,Laptop,1200
102,Keyboard,75
103,Mouse,25
104,Monitor,300"""

csv_file = io.StringIO(csv_data)

df_csv = pd.read_csv(csv_file)

display(df_csv)
```

	ID	Product	Price
0	101	Laptop	1200
1	102	Keyboard	75
2	103	Mouse	25
3	104	Monitor	300

Load datasets using scikit-learn

Show how to load built-in datasets from the scikit-learn library.

```
from sklearn.datasets import load_iris

iris = load_iris()

print(iris.DESCR)

print(iris.feature_names)

df_iris = pd.DataFrame(data=iris.data, columns=iris.feature_names)
display(df_iris.head())
```

```
.. _iris_dataset:

Iris plants dataset
--------------------

**Data Set Characteristics:**

:Number of Instances: 150 (50 in each of three classes)
:Number of Attributes: 4 numeric, predictive attributes and the class
:Attribute Information:
    - sepal length in cm
    - sepal width in cm
    - petal length in cm
    - petal width in cm
    - class:
            - Iris-Setosa
            - Iris-Versicolour
            - Iris-Virginica

:Summary Statistics:

============== ==== ==== ======= ===== ====================
                Min  Max   Mean    SD   Class Correlation
============== ==== ==== ======= ===== ====================
sepal length:   4.3  7.9   5.84   0.83    0.7826
sepal width:    2.0  4.4   3.05   0.43   -0.4194
petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)
petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)
============== ==== ==== ======= ===== ====================

:Missing Attribute Values: None
:Class Distribution: 33.3% for each of 3 classes.
:Creator: R.A. Fisher
:Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
:Date: July, 1988

The famous Iris database, first used by Sir R.A. Fisher. The dataset is taken
from Fisher's paper. Note that it's the same as in R, but not as in the UCI
Machine Learning Repository, which has two wrong data points.

This is perhaps the best known database to be found in the
pattern recognition literature.  Fisher's paper is a classic in the field and
is referenced frequently to this day.  (See Duda & Hart, for example.)  The
data set contains 3 classes of 50 instances each, where each class refers to a
type of iris plant.  One class is linearly separable from the other 2; the
latter are NOT linearly separable from each other.

.. dropdown:: References

  - Fisher, R.A. "The use of multiple measurements in taxonomic problems"
    Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to
    Mathematical Statistics" (John Wiley, NY, 1950).
  - Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Analysis.
    (Q327.D83) John Wiley & Sons.  ISBN 0-471-22361-1.  See page 218.
  - Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New System
    Structure and Classification Rule for Recognition in Partially Exposed
    Environments".  IEEE Transactions on Pattern Analysis and Machine
    Intelligence, Vol. PAMI-2, No. 1, 67-71.
  - Gates, G.W. (1972) "The Reduced Nearest Neighbor Rule".  IEEE Transactions
    on Information Theory, May 1972, 431-433.
  - See also: 1988 MLC Proceedings, 54-64.  Cheeseman et al"s AUTOCLASS II
    conceptual clustering system finds 3 classes in the data.
  - Many, many more ...

['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Compute basic statistics

Calculate mean, median, mode, variance, and standard deviation using pandas.

```
mean_salary = df1['Salary'].mean()

median_salary = df1['Salary'].median()

mode_age = df1['Age'].mode()

variance_salary = df1['Salary'].var()

std_dev_salary = df1['Salary'].std()

print(f"Mean: {mean_salary}")
print(f"Median: {median_salary}")
print(f"Mode: {mode_age.tolist()}")
print(f"Variance: {variance_salary}")
print(f"Standard Deviation: {std_dev_salary}")
```

```
Mean of Salary: 62000.0
Median of Salary: 60000.0
Mode of Age: [25, 28, 30, 32, 35]
Variance of Salary: 107500000.0
Standard Deviation of Salary: 10368.220676663861
```

Demonstrate data preprocessing

Illustrate various preprocessing techniques such as handling missing values and feature normalization.

```
df1_preprocessed = df1.copy()

df1_preprocessed.loc[2, 'Salary'] = None
df1_preprocessed.loc[4, 'Age'] = None

mean_salary_fill = df1_preprocessed['Salary'].mean()
df1_preprocessed['Salary'].fillna(mean_salary_fill, inplace=True)

median_age_fill = df1_preprocessed['Age'].median()
df1_preprocessed['Age'].fillna(median_age_fill, inplace=True)

cols_to_normalize = ['Salary', 'Age']
df_subset = df1_preprocessed[cols_to_normalize]

df_normalized = (df_subset - df_subset.min()) / (df_subset.max() - df_subset.min())

df1_preprocessed[cols_to_normalize] = df_normalized

display(df1_preprocessed)
```

```
/tmp/ipython-input-15-835128090.py:7: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting valu

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].me


  df1_preprocessed['Salary'].fillna(mean_salary_fill, inplace=True)
/tmp/ipython-input-15-835128090.py:10: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained ass
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting valu

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].me


  df1_preprocessed['Age'].fillna(median_age_fill, inplace=True)
```

	ID	Name	Age	Salary
0	1	Alice	0.0	0.0000
1	2	Bob	0.5	0.5000
2	3	Charlie	1.0	0.4375
3	4	David	0.3	0.2500
4	5	Eve	0.4	1.0000

Reshape data

Show how to reshape data using pandas.

```
data_long = {
    'Category': ['A', 'B', 'C'],
    'Value1': [10, 20, 30],
    'Value2': [100, 200, 300]
}
df_long = pd.DataFrame(data_long)

df_long_melted = pd.melt(df_long,
                         id_vars='Category',
                         value_vars=['Value1', 'Value2'],
                         var_name='Variable',
                         value_name='Value')

display(df_long_melted)
```

	Category	Variable	Value
0	A	Value1	10
1	B	Value1	20
2	C	Value1	30
3	A	Value2	100
4	B	Value2	200
5	C	Value2	300

Filter data

Demonstrate how to filter data based on conditions using pandas.

```
high_salary_df = df1[df1['Salary'] > 60000].copy()

young_high_salary_df = df1[(df1['Age'] < 30) & (df1['Salary'] >= 55000)].copy()

display("DataFrame with Salary > 60000:")
display(high_salary_df)
display("DataFrame with Age < 30 and Salary >= 55000:")
display(young_high_salary_df)
```

```
'DataFrame with Salary > 60000:'
```