```
import numpy as np
from scipy import stats

weeks = np.array([1, 2, 3, 4, 5])
sales = np.array([1.2, 1.8, 2.6, 3.2, 3.8])

coefficients = np.polyfit(weeks, sales, 1)
slope = coefficients[0]
intercept = coefficients[1]

predict_sales_7th = slope * 7 + intercept
predict_sales_12th = slope * 12 + intercept

actual_sales_7th = 4
actual_sales_12th = 5

mse_7th = (predict_sales_7th - actual_sales_7th) ** 2
mse_12th = (predict_sales_12th - actual_sales_12th) ** 2

rmse_7th = np.sqrt(mse_7th)
rmse_12th = np.sqrt(mse_12th)

mae_7th = np.abs(predict_sales_7th - actual_sales_7th)
mae_12th = np.abs(predict_sales_12th - actual_sales_12th)
sse_7th = np.sum((predict_sales_7th - actual_sales_7th) ** 2)
sse_12th = np.sum((predict_sales_12th - actual_sales_12th) ** 2)
r2_7th = 1 - (sse_7th / np.sum((actual_sales_7th - np.mean(actual_sales_7th)) **
r2_12th = 1 - (sse_12th / np.sum((actual_sales_12th - np.mean(actual_sales_12th)

print("Mean Squared Error (MSE) for 7th week: ", mse_7th)
print("Root Mean Squared Error (RMSE) for 7th week: ", rmse_7th)
print("Mean Absolute Error (MAE) for 7th week: ", mae_7th)
print("Sum of Squared Error (SSE) for 7th week: ", sse_7th)
print("R-squared Error for 7th week: ", r2_7th)

print("Mean Squared Error (MSE) for 12th week: ", mse_12th)
print("Root Mean Squared Error (RMSE) for 12th week: ", rmse_12th)
print("Mean Absolute Error (MAE) for 12th week: ", mae_12th)
print("Sum of Squared Error (SSE) for 12th week: ", sse_12th)
print("R-squared Error for 12th week: ", r2_12th)
```

```
Mean Squared Error (MSE) for 7th week:  1.3455999999999984
Root Mean Squared Error (RMSE) for 7th week:  1.1599999999999993
Mean Absolute Error (MAE) for 7th week:  1.1599999999999993
Sum of Squared Error (SSE) for 7th week:  1.3455999999999984
R-squared Error for 7th week:  -inf
Mean Squared Error (MSE) for 12th week:  11.971599999999993
Root Mean Squared Error (RMSE) for 12th week:  3.459999999999999
Mean Absolute Error (MAE) for 12th week:  3.459999999999999
Sum of Squared Error (SSE) for 12th week:  11.971599999999993
R-squared Error for 12th week:  -inf
<ipython-input-4-d2d94d856a4b>:38: RuntimeWarning: divide by zero encounter
  r2_7th = 1 - (sse_7th / np.sum((actual_sales_7th - np.mean(actual_sales_7
<ipython-input-4-d2d94d856a4b>:39: RuntimeWarning: divide by zero encounter
```

```
       r2_12th = 1 - (sse_12th / np.sum((actual_sales_12th - np.mean(actual_sale
```

```
import numpy as np
from scipy import stats

# Replace the following line with your own data
data = np.array([23, 45, 56, 67, 78, 21, 45, 56, 67, 99])

mean = np.mean(data)
median = np.median(data)
std_dev = np.std(data)
variance = np.var(data)
mode_result = stats.mode(data)

print(f"Mean: {mean}")
print(f"Median: {median}")
print(f"Standard deviation: {std_dev}")
print(f"Variance: {variance}")
print(f"Mode: {mode_result.mode}")
```

```
    Mean: 55.7
    Median: 56.0
    Standard deviation: 22.649724060129298
    Variance: 513.01
    Mode: 45
```

```
from sklearn.neighbors import KNeighborsClassifier
import numpy as np

# data set
X = np.array([[2, 3.5], [5, 5.5], [3.5, 4.5], [4.5, 4.5], [4.5, 2.5], [2.5, 6.5
Y = np.array([2, 3, 4, 2, 6, 4, 5, 2, 3, 2, 7, 7, 4, 6, 3])

# input for classification
input_point = np.array([[5.5, 5.5]])

# instantiate the KNN classifier with 3 nearest neighbors
knn = KNeighborsClassifier(n_neighbors=3)

# fit the classifier to the data
knn.fit(X, Y)

# predict the class of the input point
predicted_class = knn.predict(input_point)

print("The predicted class of the input point (5.5, 5.5) is:", predicted_class[
```

```
    The predicted class of the input point (5.5, 5.5) is: 3
```