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

import warnings
warnings.filterwarnings('ignore')
```

```
!wget --no-check-certificate https://drive.google.com/uc?id=15B0YXLJhx41faA2rVyifWMjlmpuFqAxc -O loan.csv
```

```
--2026-01-28 15:14:52--  https://drive.google.com/uc?id=15B0YXLJhx41faA2rVyifWMjlmpuFqAxc
Resolving drive.google.com (drive.google.com)... 108.177.97.113, 108.177.97.139, 108.177.97.102, ...
Connecting to drive.google.com (drive.google.com)|108.177.97.113|:443... connected.
HTTP request sent, awaiting response... 303 See Other
Location: https://drive.usercontent.google.com/download?id=15B0YXLJhx41faA2rVyifWMjlmpuFqAxc [following]
--2026-01-28 15:14:52--  https://drive.usercontent.google.com/download?id=15B0YXLJhx41faA2rVyifWMjlmpuFqAxc
Resolving drive.usercontent.google.com (drive.usercontent.google.com)... 74.125.23.132, 2404:6800:4008:c02::84
Connecting to drive.usercontent.google.com (drive.usercontent.google.com)|74.125.23.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 38011 (37K) [application/octet-stream]
Saving to: 'loan.csv'

loan.csv            100%[===================>]  37.12K  --.-KB/s    in 0s

2026-01-28 15:14:54 (90.9 MB/s) - 'loan.csv' saved [38011/38011]
```

```
data = pd.read_csv('loan.csv')
data.shape
```

```
(614, 13)
```

One hot encoding

```
pd.get_dummies(data["Gender"]).astype(int)
```

	Female	Male
0	0	1
1	0	1
2	0	1
3	0	1
4	0	1
...	...	...
609	1	0
610	0	1
611	0	1
612	0	1
613	1	0

614 rows × 2 columns

```
from sklearn.preprocessing import LabelEncoder
```

```
label_encoder=LabelEncoder()
col="Gender"
data[col]=label_encoder.fit_transform(data[col])
data[col].value_counts()
```

Gender	count
1	489
0	112
2	13

dtype: int64

```
dummy_df = pd.DataFrame({
    'Gender': ['Male', 'Female', 'Female', 'Male', 'Male', 'Female'],
    'Married': ['Yes', 'No', 'Yes', 'Yes', 'No', 'No'],
    'Loan_Status': [1, 0, 1, 1, 0, 0]  # Target variable
})

dummy_df
```

	Gender	Married	Loan_Status
0	Male	Yes	1
1	Female	No	0
2	Female	Yes	1
3	Male	Yes	1
4	Male	No	0
5	Female	No	0

Next steps: Generate code with dummy_df | New interactive sheet

```
target='Loan_Status'
categorica_col='Gender'
```

```
encode_map=dummy_df.groupby(categorica_col)[target].mean()
encode_map
```

Gender	Loan_Status
Female	0.333333
Male	0.666667

dtype: float64

```
dummy_df["Gender"].map(encode_map)
```

	Gender
0	0.666667
1	0.333333
2	0.333333
3	0.666667
4	0.666667
5	0.333333

dtype: float64

```
!pip install category_encoders
```

```
Collecting category_encoders
  Downloading category_encoders-2.9.0-py3-none-any.whl.metadata (7.9 kB)
Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.12/dist-packages (from category_encoders)
Requirement already satisfied: pandas>=1.0.5 in /usr/local/lib/python3.12/dist-packages (from category_encoders)
Requirement already satisfied: patsy>=0.5.1 in /usr/local/lib/python3.12/dist-packages (from category_encoders) (
Requirement already satisfied: scikit-learn>=1.6.0 in /usr/local/lib/python3.12/dist-packages (from category_enco
Requirement already satisfied: scipy>=1.0.0 in /usr/local/lib/python3.12/dist-packages (from category_encoders) (
Requirement already satisfied: statsmodels>=0.9.0 in /usr/local/lib/python3.12/dist-packages (from category_encod
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas>=1.
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas>=1.0.5->categ
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas>=1.0.5->cat
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn>=1.6.0
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.12/dist-packages (from statsmodels>=0.9.
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->
Downloading category_encoders-2.9.0-py3-none-any.whl (85 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 85.9/85.9 kB 3.0 MB/s eta 0:00:00
Installing collected packages: category_encoders
Successfully installed category_encoders-2.9.0
```

```
from category_encoders import TargetEncoder
```

```
dummy_df
```

	Gender	Married	Loan_Status
0	Male	Yes	1
1	Female	No	0
2	Female	Yes	1
3	Male	Yes	1
4	Male	No	0
5	Female	No	0

Next steps: Generate code with dummy_df | New interactive sheet

```
encoder=TargetEncoder(cols=["Gender"])
dummy_df["gender_target_encoded"]=encoder.fit_transform(dummy_df["Gender"],dummy_df["Loan_Status"])
```

```
dummy_df[["Gender","gender_target_encoded"]]
```

	Gender	gender_target_encoded
0	Male	0.525744
1	Female	0.474256
2	Female	0.474256
3	Male	0.525744
4	Male	0.525744
5	Female	0.474256

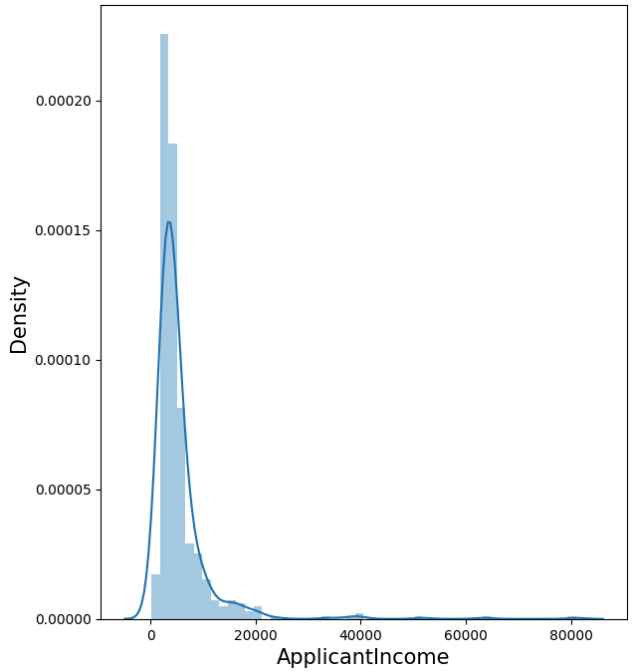
Feature Scaling (Standardization)

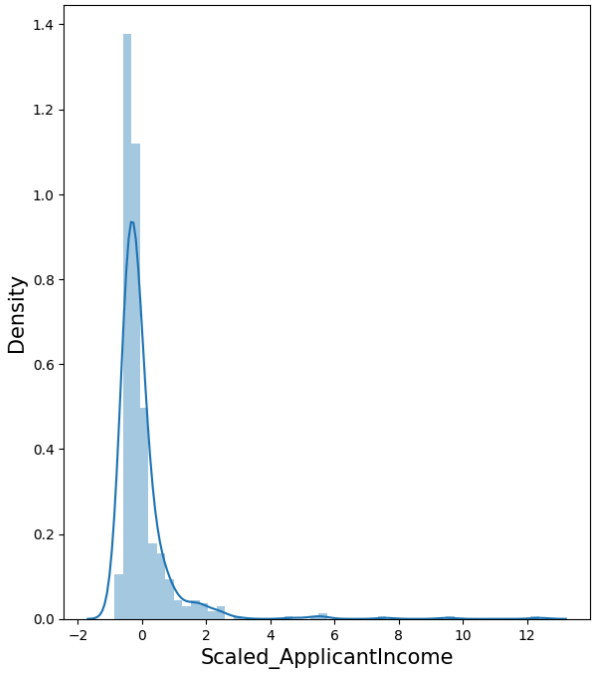
```
from sklearn.preprocessing import StandardScaler
```

```
scaler=StandardScaler()
```

```
data["Scaler_ApplicantIncome"]=scaler.fit_transform(data[["ApplicantIncome"]])
```

```
# set the figure size
plt.rcParams["figure.figsize"] = [15,8]

# subplot is used to create a set of plots
# we create a subplot of 1 row by 2 columns
# use first column for plotting
plt.subplot(1,2,1)

# plot the distribution of the original variable
# axlabel: label the x-axis
sns.distplot(data["ApplicantIncome"])
# set label for the y-axis
plt.ylabel('Density', fontsize=15)
# set label for x-axis
plt.xlabel('ApplicantIncome', fontsize=15)

# We create a subplot of 1 row by 2 columns
# use the second column for plotting
plt.subplot(1,2,2)

# plot the distribution of the scaled variable
# axlabel: label the x-axis
sns.distplot(data["Scaler_ApplicantIncome"])
# set label for the y-axis
plt.ylabel('Density', fontsize=15)
# set label for the x-axis
plt.xlabel('Scaled_ApplicantIncome', fontsize=15)
# display the plot
plt.show()
```





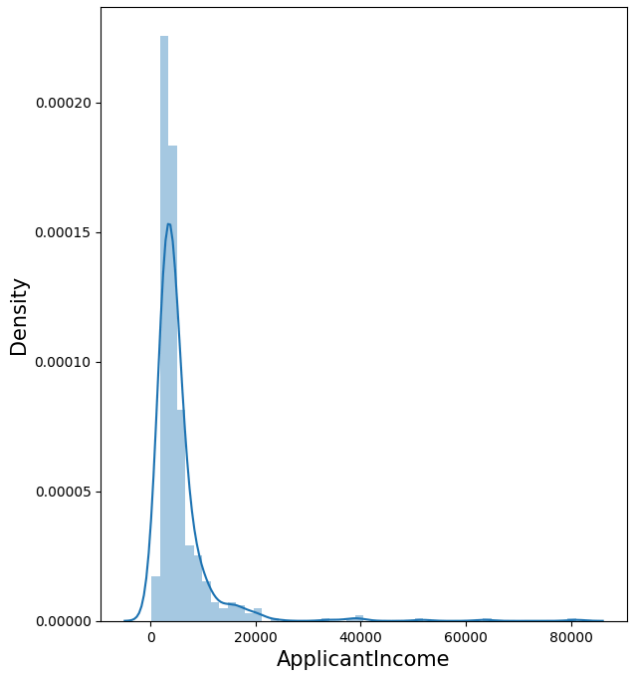
Normalization

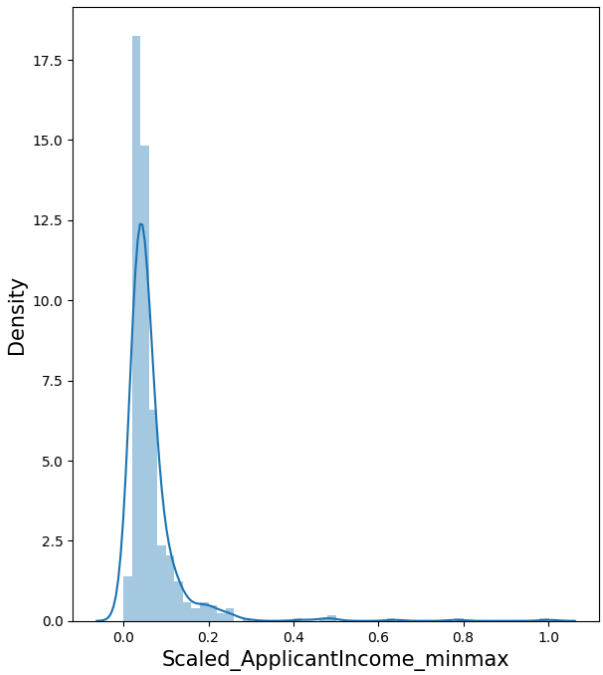
```
from sklearn.preprocessing import MinMaxScaler
min_max= MinMaxScaler()
```

```
data["Scaler_ApplicantIncome_minmax"]=min_max.fit_transform(data[["ApplicantIncome"]])
```

```
data["Scaler_LoanAmount_minmax"]=min_max.fit_transform(data[["LoanAmount"]])
```

```
# set the figure size
plt.rcParams["figure.figsize"] = [15,8]

# subplot is used to create a set of plots
# we create a subplot of 1 row by 2 columns
# use first column for plotting
plt.subplot(1,2,1)

# plot the distribution of the original variable
# axlabel: label the x-axis
sns.distplot(data["ApplicantIncome"])
# set label for the y-axis
plt.ylabel('Density', fontsize=15)
# set label for x-axis
plt.xlabel('ApplicantIncome', fontsize=15)

# We create a subplot of 1 row by 2 columns
# use the second column for plotting
plt.subplot(1,2,2)

# plot the distribution of the scaled variable
# axlabel: label the x-axis
sns.distplot(data["Scaler_ApplicantIncome_minmax"])
# set label for the y-axis
plt.ylabel('Density', fontsize=15)
# set label for the x-axis
plt.xlabel('Scaled_ApplicantIncome_minmax', fontsize=15)
# display the plot
plt.show()
```






```
data["Scaler_LoanAmount_minmax"].min(),data["Scaler_LoanAmount_minmax"].max()
```

```
(0.0, 1.0)
```

PS-2

Question 1

A Data Analyst at a retail company wants to analyse the average customer satisfaction scores (out of 10) across three store locations (A, B, and C) to check if there is any significant difference in satisfaction scores between the stores. The sample data is as follows:

- Store A = [8.2, 8.5, 7.9, 8.1, 8.3, 8.6]
- Store B = [7.4, 7.8, 7.5, 7.3, 7.6, 7.7]
- Store C = [8.8, 8.7, 9.0, 8.9, 8.6, 8.5]

The analyst:

1. Checks for normality using the Shapiro-Wilk test.
2. Uses Levene's test to check for equality of variances.
3. Performs One-Way ANOVA if assumptions are met, else performs Kruskal-Wallis Test.
4. If ANOVA is significant, performs pairwise t-tests to identify the specific pairs that differ.
5. Significance level: 0.05

What is the correct conclusion? Options:

- A. Satisfaction scores do not differ significantly across all three stores.
- B. Assumptions passed; hence, One-Way ANOVA was performed.
- C. Store A, Store B and Store C differ significantly in customer satisfaction.
- D. Levene's test passed, Shapiro-Wilk failed, so One-way ANOVA was not valid — Kruskal-Wallis was used instead.

```
# Null Hypothesis : There are no significant differences in mean satisfaction scores across the three stores.
# Alternative Hypothesis: At least one store has a significantly different mean satisfaction score.
```

```
from scipy.stats import shapiro

# Sample data: Satisfaction scores
Store_A = [8.2, 8.5, 7.9, 8.1, 8.3, 8.6]
Store_B = [7.4, 7.8, 7.5, 7.3, 7.6, 7.7]
```

```
Store_C = [8.8, 8.7, 9.0, 8.9, 8.6, 8.5]
stat_a,p_value_a=shapiro  (Store_A)
stat_b,p_value_b=shapiro (Store_B)
stat_c,p_value_c=shapiro  (Store_C)
print("StoreA:",p_value_a,"; Normally distributed" if p_value_a>0.05 else "Not Normally distributed")
print("StoreB:",p_value_b,"; Normally distributed" if p_value_b>0.05 else "Not Normally distributed")
print("StoreC:",p_value_c,"; Normally distributed" if p_value_c>0.05 else "Not Normally distributed")
```

```
StoreA: 0.9453057822048255 ; Normally distributed
StoreB: 0.9605549608523386 ; Normally distributed
StoreC: 0.960554960852335 ; Normally distributed
```

```
from scipy.stats import levene

# Levene's test for equality of variances
stat_levene, pvalue_levene = levene(Store_A, Store_B, Store_C)
print('Levene test p-value:', pvalue_levene)

if pvalue_levene < 0.05:
   print("Variances are not equal")
else:
   print("Variances are equal")
```

```
Levene test p-value: 0.6591699521904859
Variances are equal
```

```
from scipy.stats import f_oneway

# Perform One-Way ANOVA
f_stat, p_value = f_oneway(Store_A, Store_B, Store_C)

print("ANOVA Result:")
print("F-statistic:", f_stat)
print("P-value:", p_value)

alpha = 0.05
if p_value < alpha:
   print("\nReject H₀ → At least one store has significantly different customer satisfaction scores")
else:
   print("\nFail to reject H₀ → No significant difference in customer satisfaction scores")
```

```
ANOVA Result:
F-statistic: 48.012195121951265
P-value: 3.0202748614213096e-07

Reject H₀ → At least one store has significantly different customer satisfaction scores
```

```
from scipy.stats import ttest_ind

# Pairwise t-tests
result_ab = ttest_ind(Store_A, Store_B)
result_ac = ttest_ind(Store_A, Store_C)
result_bc = ttest_ind(Store_B, Store_C)

# Print results
alpha = 0.05
print("Pairwise t-test results:")
print("A vs. B:", "Pvalue = ", result_ab.pvalue, " ; Significant difference" if result_ab.pvalue <= alpha else "
print("A vs. C:", "Pvalue = ", result_ac.pvalue, " ; Significant difference" if result_ac.pvalue <= alpha else "
print("B vs. C:", "Pvalue = ", result_bc.pvalue, " ; Significant difference" if result_bc.pvalue <= alpha else "
```

```
Pairwise t-test results:
A vs. B: Pvalue =  0.0002597459072840726  ; Significant difference
A vs. C: Pvalue =  0.004020920524773078  ; Significant difference
B vs. C: Pvalue =  6.0100883931906 94e-07  ; Significant difference
```

Question 3

A Data Scientist at a financial institution is analyzing the distribution of monthly income for two different departments: **Sales** and **Marketing**.
The data collected represents the monthly incomes (in thousands of dollars) for a sample of employees from each department.

The data is as follows:

- **Sales Department**: [38, 45, 42, 40, 35, 50, 41, 43, 47, 46]
- **Marketing Department**: [50, 55, 60, 52, 57, 54, 61, 59, 63, 62]

The Data Scientist wants to test if the monthly income distributions of the two departments are different or not.
The significance level (\alpha = 0.05).

What is the correct conclusion based on the results of the KS test for comparing the distributions of the two groups?

Options:

A. The distributions of monthly income in Sales and Marketing departments are significantly different.
B. The distributions of monthly income in Sales and Marketing departments are not significantly different.
C. KS test is not appropriate for this problem because the sample sizes are different.
D. KS test is valid, but a paired t-test should be performed instead of KS.

Null Hypothesis: The two samples are drawn from the same distribution.

Alternative Hypothesis: The two samples are drawn from different distributions.

```
import numpy as np
from scipy.stats import ks_2samp
```

```
# Data for monthly income (in thousands of dollars) for each department
sales = [38, 45, 42, 40, 35, 50, 41, 43, 47, 46]
marketing = [50, 55, 60, 52, 57, 54, 61, 59, 63, 62]

# Perform the Kolmogorov-Smirnov test for two samples
statistic, p_value = ks_2samp(sales, marketing)

# Output the result
print("KS statistic:", statistic)
print("KS p-value:", p_value)

alpha = 0.05

# Conclusion
if p_value < alpha:
   print("Reject the null hypothesis. Monthly income distributions in the Sales and Marketing departments are si
else:
   print("Fail to reject the null hypothesis. Monthly income distributions in the Sales and Marketing department
```

```
KS statistic: 0.9
KS p-value: 0.00021650176448938054
Reject the null hypothesis. Monthly income distributions in the Sales and Marketing departments are significantly
```

Question 4

A financial analyst is examining the relationship between the returns of two stocks, **Stock A** and **Stock B**, over the past 10 days.
The daily returns (in percentage) for the two stocks are as follows:

- **Stock A Returns**: [1.2, 1.5, 1.7, 1.6, 1.8, 1.5, 1.6, 1.7, 1.4, 1.5]
- **Stock B Returns**: [0.9, 1.0, 1.2, 1.3, 1.4, 1.2, 1.3, 1.4, 1.1, 1.2]

The analyst wants to determine the relationship between the returns of Stock A and Stock B and assess whether the returns of the two stocks are positively correlated, negatively correlated, or uncorrelated.

Which of the following conclusions can the analyst draw based on the covariance and correlation between the two stocks' returns?

Options:

A. Since the covariance between Stock A and Stock B is positive, the returns of the two stocks are negatively correlated, and as such, they move in the same direction.
B. The correlation between Stock A and Stock B is 0, indicating no relationship between the two stocks' returns.
C. The covariance of the two stocks' returns is 0, implying there is no relationship between the returns of the two stocks.
D. The correlation between Stock A and Stock B is positive and less than 1, which indicates a moderate positive relationship between the two stocks' returns.

```
# Data for returns of Stock A and Stock B
stock_A_returns = [1.2, 1.5, 1.7, 1.6, 1.8, 1.5, 1.6, 1.7, 1.4, 1.5]
stock_B_returns = [0.9, 1.0, 1.2, 1.3, 1.4, 1.2, 1.3, 1.4, 1.1, 1.2]
```

```
import numpy as np
```

```
covariance=np.cov(stock_A_returns,stock_B_returns)[0,1]
covariance
```

```
np.float64(0.024444444444444442)
```

```
correlation=np.corrcoef(stock_A_returns,stock_B_returns)[0,1]
correlation
```

```
np.float64(0.8723567442899585)
```

A Digital Marketing Manager at an e-commerce company is running an A/B test to compare two versions of an email marketing campaign. Version A (control) and Version B (treatment) are sent to different user groups, and the goal is to determine if the new design (Version B) increases the click-through rate (CTR) more effectively than the current design (Version A). The CTR data (in percentage) for both versions is as follows:

Version A (Control): [2.1, 2.3, 2.4, 2.5, 2.2, 2.0, 2.3, 2.1, 2.2, 2.4] Version B (Treatment): [2.8, 3.0, 2.9, 3.1, 2.7, 3.0, 2.9, 3.2, 2.8, 3.1]

The manager wants to test whether Version B has a significantly higher click-through rate than Version A. The significance level is set to 0.05.

What is the correct conclusion based on the A/B test?

Options:

A. Version B performs significantly better than Version A in terms of click-through rate.

B. There is no significant difference between Version A and Version B in terms of click-through rate.

C. The A/B test is invalid because the sample size is too small.

D. A two-sample t-test is not appropriate for A/B testing. A paired t-test should be used instead.

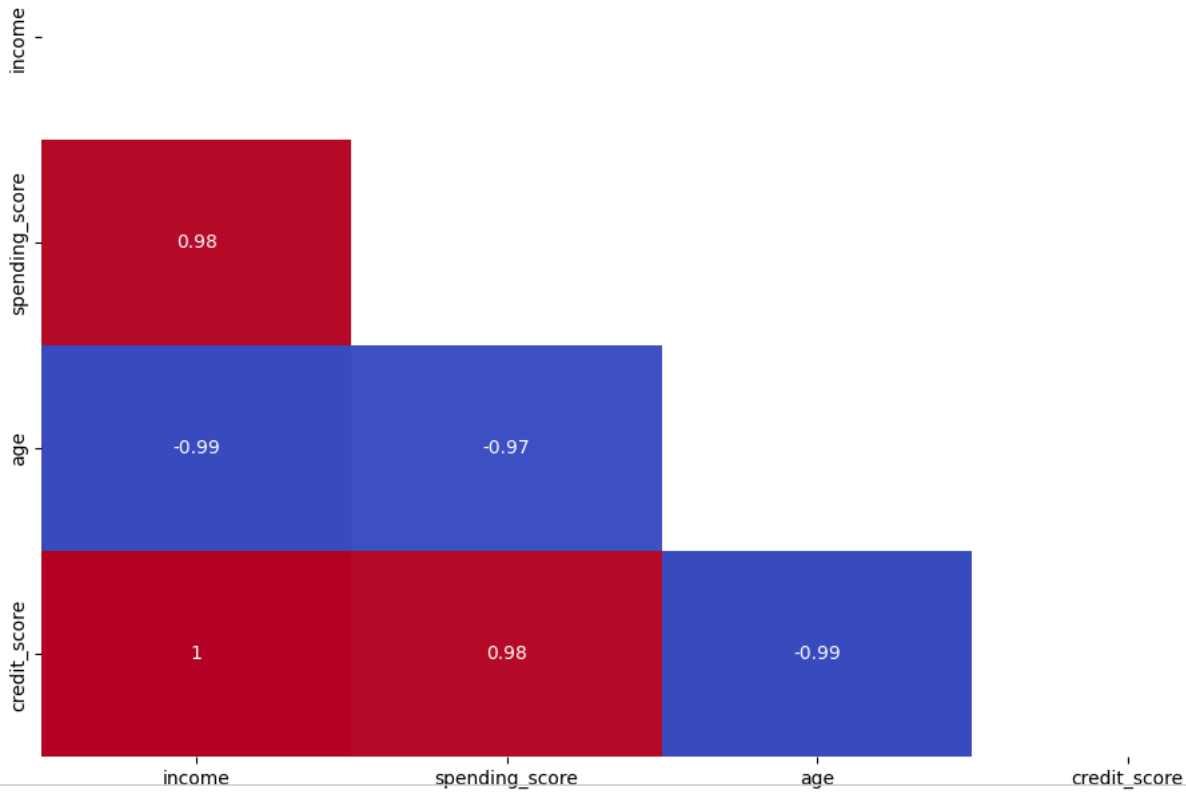
H0: No significant difference

Ha: Significant difference

```
import numpy as np
from scipy.stats import ttest_ind

# Data for click-through rate (CTR) for both versions
version_A = [2.1, 2.3, 2.4, 2.5, 2.2, 2.0, 2.3, 2.1, 2.2, 2.4]
version_B = [2.8, 3.0, 2.9, 3.1, 2.7, 3.0, 2.9, 3.2, 2.8, 3.1]

# Perform two-sample t-test (assuming unequal variances)
statistic, p_value = ttest_ind(version_B, version_A, alternative= 'greater',equal_var=False)

# Output the result
print("T-statistic:", statistic)
print("P-value:", p_value)

alpha = 0.05

# Conclusion
if p_value < alpha:
   print("Reject the null hypothesis. Version B has a significantly higher click-through rate than Version A.")
else:
   print("Fail to reject the null hypothesis. Version B has no significant difference in click-through rate than
```

```
T-statistic: 9.899494936611667
P-value: 5.2153315601116904e-09
Reject the null hypothesis. Version B has a significantly higher click-through rate than Version A.
```

```
import seaborn as sns
import matplotlib.pyplot as plt
# Example dataset
df = pd.DataFrame({
   'income': [4000, 5000, 6000, 7000],
   'spending_score': [20, 30, 70, 90],
   'age': [50, 40, 30, 25],
   'credit_score': [600, 650, 700, 750]
})

corr = df.corr()
sns.heatmap(corr, annot=True, cmap="coolwarm", mask=np.triu(np.ones_like(corr, dtype=bool)))
plt.show()
```



Which of the following conclusions is most valid and supported by the correlation matrix?

Options:

A. Customers with higher credit scores tend to be older and have lower income and spending.

B. Younger customers with higher income tend to have lower credit scores.

C. Customers with higher income and spending score tend to have higher credit_score, and all three variables are negatively correlated with age.

D. Age and income are positively correlated, but both negatively affect the spending_score

```
Start coding or generate with AI.
```