

INFERENTIAL STATISTICS & LINEAR REGRESSION

Module Code: B8IT152

Names: Jack Maguire; Sean Carroll; Shane Crowley

Student Numbers: 20026112; 20024157; 20024425

Group: 2

Question 1 (a)

Consider two variables of the dataset, and develop a decision-making strategy to check whether two averages of variables are equal at the significant level $\alpha=0.01$.

The null hypothesis is that one column is equal to another column. The alternative hypothesis is that one column does not equal the other. The test score is calculated using sample mean (\bar{x}), variance (segmasq), and sample size (n). The C-score was calculated using a significant level of 0.01 ($1 - \alpha/2$) as a two-sided hypothesis). The test score was less than the c-value therefore H_0 was accepted.

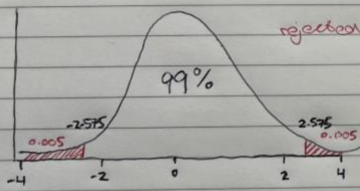
Question 1 (A)

(i) State the Hypothesis
 $H_0: x = y$
 $H_1: x \neq y$

(ii) Find the statistic score + critical values
- significant level: $\alpha(\alpha) = 0.01$
- test score:

$$\frac{(\bar{x}_1 - \bar{x}_2)}{\sqrt{(\sigma_1^2/n_1) + (\sigma_2^2/n_2)}}$$
$$= \frac{(50.3584 - 50.2763)}{\sqrt{(830.4019/10000) + (834.5842/10000)}}$$
$$= 0.201$$

(iii) Find C-value:
 $= \text{qnorm}(1 - \alpha/2)$



C-value = 2.575

(iv) Test score < C-value $\therefore H_0$ is Accepted.

Python Code to calculate whether the null hypothesis is accepted or rejected.

```
# Import read_csv from pandas
from pandas import read_csv

# Call the file data
data = read_csv('/content/ParisHousing.csv')

# Import the required modules and functions
from statistics import mean, variance
from math import sqrt
from scipy.stats import norm

# TWO-SIDED HYPOTHESIS

# x = numberOfRooms
# y = floors
x = data['numberOfRooms']
y = data['floors']

# i. State the hypotheses.
# H0: x = y
# H1: x != y

# ii. Find the statistic (test) and critical values.
alpha = 0.01

xbar1 = mean(x)
xbar2 = mean(y)

segmasq1 = variance(x)
segmasq2 = variance(y)

n1 = len(x)
n2 = len(y)
```

```
# calculate test score
test_score = (xbar1 - xbar2) / sqrt (segmasq1/n1 + segmasq2/n2)

# calculate c value
c_value = norm.ppf(1 - alpha/2)

# iii. Explain your decision and Interpret the results.
print(test_score, c_value)

print('')

# Test_value > c_value. Therefore H0 is accepted.
print('Test score of', test_score, 'is less than c value of', c_value, 'therefore H0 is accepted.')
```

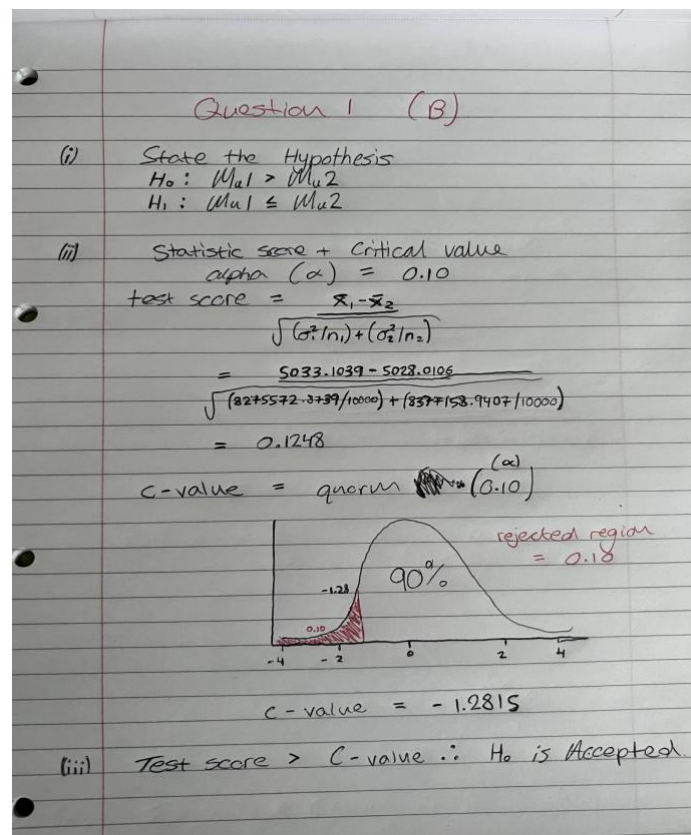
```
0.20120456835054987 2.5758293035489004
```

Test score of 0.20120456835054987 is less than c value of 2.5758293035489004 therefore H0 is accepted.

Question 1 (b)

Consider two variables of the dataset, and develop a decision-making strategy to check whether two averages of variables are different at the significant level $\alpha=0.10$.

The null hypothesis is that one column (basement) is more than another column (attic). The alternative hypothesis is that the column 'basement' is less than or equal to the 'attic' column. The test score is calculated using sample mean (\bar{x}), variance (segmasq), and sample size (n). The c-value is calculated using the significant value of 0.10 (upper sided hypothesis). The test score is greater than the c-value therefore H_0 was accepted.



Python Code to calculate whether the null hypothesis is accepted or rejected.

```
# DIFFERENCE so one column less/ more than
# Chosen Difference: More than

# To see if the average of column 'basement' is MORE THAN 'attic'
# x2 = basement
# y2 = attic
x2 = data['basement']
y2 = data['attic']

# i. State the hypotheses.
# H0:  $\mu_1 > \mu_2$ 
# H1:  $\mu_1 \leq \mu_2$ 

# ii. Find the statistic (test) and critical values.
alpha2 = 0.10
xbar3 = mean(x2)
xbar4 = mean(y2)

segmasq3 = variance(x2)
segmasq4 = variance(y2)

n3 = len(x2)
n4 = len(y2)

# calculate test score
test_score2 = (xbar3 - xbar4) / sqrt (segmasq3/n3 + segmasq4/n4)

# calculate c value
c_value2 = norm.ppf(alpha2)

# iii. Explain your decision and Interpret the results.
print(test_score2, c_value2)

print('')
```

```
# Test_value > c_value. Therefore H0 is accepted.
print('Test score of', test_score2, 'is greater than the c value of', c_value2, 'therefore H0 is accepted.')
```

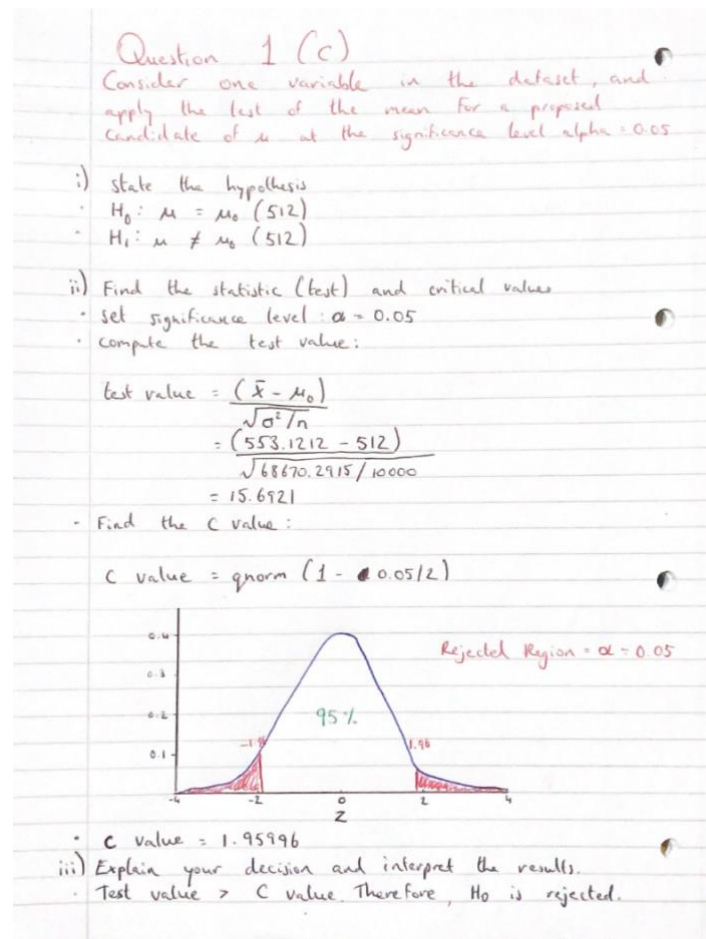
```
0.12481204910251581 -1.2815515655446004
```

Test score of 0.12481204910251581 is greater than the c value of -1.2815515655446004 therefore H0 is accepted.

Question 1 (c)

Consider one variable in the dataset, and apply the test of the mean for a proposed candidate of μ at the significant level $\alpha=0.05$.

The null hypothesis (H_0) is that the population mean is equal to 512. The alternative hypothesis (H_1) is that the population mean is not equal to 512. The test statistic (test value) and a critical value (c value). The test statistic is calculated using the sample mean (\bar{x}), the assumed population variance (σ^2), and the sample size (n). Once the c value is calculated, with the significance level α (0.05 in this case) the null hypothesis was rejected as the test value was greater than the c value.



Python Code to calculate whether the null hypothesis is accepted or rejected.

```
# To see if the average of column 'garage' is equal to 512 (Mu0)
# x3 = garage
x3 = data['garage']

[53] # i. State the hypotheses.
# H0: Mu = 512
# H1: Mu != 512

# ii. Find the statistic (test) and critical values.
Mu0 = 512

alpha3 = 0.05

xbar5 = mean(x3)

sigmasq = variance(x3)

n5 = len(x3)

# calculate test score
test_score3 = (xbar5 - Mu0) / sqrt(sigmasq / n5)

# calculate c value
c_value3 = norm.ppf(1-alpha3/2)

# iii. Explain your decision and Interpret the results.
print(test_score3, c_value3)

print('')

# Test_value > c_value. Therefore H0 is rejected.
print('Test score of', test_score3, 'is greater than c value of', c_value3, 'therefore H0 is rejected.')

15.69210965104916 1.959963984540054

Test score of 15.69210965104916 is greater than c value of 1.959963984540054 therefore H0 is rejected.
```

Question 2 (a)

Build an ordinary least square model (OLS) for your dataset and show the summary information.

The code below uses the pandas, numpy, and statsmodels libraries to perform ordinary least squares (OLS) linear regression analysis on the Paris Housing dataset. The dataset is loaded into a pandas DataFrame, and the target variable ('price') and a set of feature variables ('squareMeters', 'numberOfRooms', 'hasYard', 'hasPool', 'isNewBuilt') are extracted. A constant term is added to the feature variables, and an OLS regression model is created using the statsmodels library. This code essentially conducts a linear regression analysis to explore the relationships between housing prices and various features in the dataset.


```

import pandas as pd
import numpy as np
import statsmodels.api as sm

# Load the dataset
df = pd.read_csv('ParisHousing.csv')

# Separate the target variable (y) and feature variables (x)
y = df['price']
x = df[['squareMeters', 'numberOfRooms', 'hasYard', 'hasPool', 'isNewBuilt']]

# Add constant to independent variables
x = sm.add_constant(x)

# Create an OLS model
model = sm.OLS(y, x)

# Fit the model
results = model.fit()

# Print the summary statistics of the fitted model
print(model.fit().summary())

```

```

=====
                        OLS Regression Results
=====
Dep. Variable:          price      R-squared:          1.000
Model:                  OLS        Adj. R-squared:       1.000
Method:                 Least Squares   F-statistic:      2.711e+09
Date:                   Wed, 20 Dec 2023   Prob (F-statistic): 0.00
Time:                   13:39:23         Log-Likelihood:   -92311.
No. Observations:       10000          AIC:             1.846e+05
Df Residuals:           9994          BIC:             1.847e+05
Df Model:                5
Covariance Type:        nonrobust
=====

```

	coef	std err	t	P> t	[0.025	0.975]
const	3263.4642	78.450	41.599	0.000	3109.687	3417.242
squareMeters	100.0001	0.001	1.16e+05	0.000	99.998	100.002
numberOfRooms	1.5239	0.858	1.777	0.076	-0.158	3.205
hasYard	3010.4435	49.442	60.889	0.000	2913.527	3107.360
hasPool	2967.3858	49.437	60.023	0.000	2870.478	3064.293
isNewBuilt	165.3642	49.427	3.346	0.001	68.477	262.252

```

=====
Omnibus:                 577.489   Durbin-Watson:           2.024
Prob(Omnibus):            0.000   Jarque-Bera (JB):        681.941
Skew:                     0.638   Prob(JB):                8.29e-149
Kurtosis:                 3.104   Cond. No.                1.97e+05
=====

```

Notes:

- [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.97e+05. This might indicate that there are strong multicollinearity or other numerical problems.

Question 2 (b)

Find the intercept and the coefficients, and write the model equation that considers only significant features.

The code below builds a linear regression model equation, where the predicted housing price is calculated by multiplying each feature's coefficient with its corresponding value in the dataset and summing them up along with the constant term. This equation represents the model's prediction for housing prices based on the given set of coefficients and features.

```
# Coefficients from the regression results
const_coef = 3263.4642
squareMeters_coef = 100.0001
numberOfRooms_coef = 1.5239
hasYard_coef = 3010.4435
hasPool_coef = 2967.3858
isNewBuilt_coef = 165.3642

# Intercept
intercept = 3263.4642

# Write the model equation
predicted_price = (const_coef + (squareMeters_coef * x['squareMeters']) +
                  (numberOfRooms_coef * x['numberOfRooms']) +
                  (hasYard_coef * x['hasYard']) +
                  (hasPool_coef * x['hasPool']) +
                  (isNewBuilt_coef * x['isNewBuilt']))
```

Question 2 (c)

Evaluate your model using an appropriate evaluation metric and find the residuals for each data point.

The code below assesses the performance and statistical significance of the linear regression model. It prints the R-squared value, which measures the proportion of the response variable's variance explained by the independent variables. A high R-squared close to 1 suggests that the independent variables effectively explain the variability in the dependent variable. The code also prints the p-values associated with each coefficient, indicating their statistical significance.

Coefficients with p-values close to 0 are considered statistically significant. In this case, all coefficients have p-values close to 0, indicating high statistical significance. However, we see that the number of rooms has a p-value slightly above the common significance level of 0.05, suggesting it may not be statistically significant at that threshold. Finally, the residuals, which represent the differences between the observed and predicted values, are printed to assess the model's goodness of fit.

```

# Evaluate the model

# Access R-squared from the results
print("R-squared:", results.rsquared)

# R-Squared result is very close to 1 indicating the independant variables
# explain almost all the variability in the dependant variable.

# Access P value from results
print('pValue:', results.pvalues)

# All coefficients are very close to 0 indicating a high level of
# statistical significance. Number of rooms has a p-value of 0.075681
# which is above the common significance level of 0.05 suggesting that number of rooms
# may not be statistically significant at 0.05 level

# Access Residuals from results
print('residual:', results.resid)

```

```

R-squared: 0.9999992628387301
pValue: const      0.000000
squareMeters      0.000000
numberOfRooms     0.075681
hasYard           0.000000
hasPool           0.000000
isNewBuilt        0.000824
dtype: float64
residual: 0      536.903594
1      -586.401221
2      -2883.902948
3      -2377.812289
4       2707.832688
...
9995    -2540.784783
9996    -1116.880159
9997     2486.914693
9998    -1870.307468
9999    -848.609064
Length: 10000, dtype: float64

```

Question 2 (d)

Predict the target values for the last 10 rows in your dataset, and predict one more value from your suggestion.

The code below predicts housing prices using the fitted regression model. It first predicts the prices for the last 10 rows of the dataset using the 'predict' method from the regression results. Additionally, the code predicts the housing price for a specific set of input values we used as an example (squareMeters=3000, numberOfRooms=9, hasYard=1, hasPool=1, isNewBuilt=0).

```
# Predict the price of last 10 rows
price_prediction = results.predict(x.tail(10))

# Set the display options to show the full numbers without scientific notation
pd.set_option('display.float_format', lambda x: '%.3f' % x)

# Print the predicted prices for the last 10 rows
print("Predicted Prices for the Last 10 Rows:")

print(price_prediction)

# Predict one more value
additional_prediction = (
    const_coef +
    (squareMeters_coef * 3000) +
    (numberOfRooms_coef * 9) +
    (hasYard_coef * 1) +
    (hasPool_coef * 1) +
    (isNewBuilt_coef * 0)
)

print("\nPredicted Price for Additional Input:")
print(additional_prediction)
```

Predicted Prices for the Last 10 Rows:

```
9990    560030.575
9991    9623769.242
9992    3360984.103
9993     37655.352
9994    2157641.083
9995     178966.685
9996    4449590.880
9997    8387543.585
9998    5906977.307
9999    147557.009
dtype: float64
```

Predicted Price for Additional Input:
309255.30859999993