

MATHEMATICS AND STATISTICS ASSIGNMENT COVER SHEET

Student to Complete

Name:	Christian Alexander Gilser
Degree Programme:	MSc. Applied Statistics
Module:	STATISTICAL ANALYSIS
Lecturer:	MARSHALL
Assignment Title/Number:	ASSIGNMENT TWO (Spring)
Due Date:	Thursday, 30 th April, 2020

Declaration

I hereby testify that, unless otherwise acknowledged, the work submitted here is entirely my own

Signature:	
------------	---

LEAVE BLANK – OFFICIAL USE ONLY

Date Received:

(Unscaled) Mark/Grade	Late – up to 14 days	Late – more than 14 days

CHRISTIAN GILSON - GLM & MULTIVARIATE STATISTICAL ANALYSIS ASSIGNMENT

April 30, 2020

1 Q1)

Study on infant respiratory disease

Dependent variable: binary category (1/0 flag) for infants on whether or not they developed the disease in the first year of their life (this is the dependent variable)

Explanatory variables: **(1)** genders of each infant (binary category boy/girl); **(2)** how fed (three categories: breast-fed, bottle-fed, supplement)

Fitted model in R:

```
fit <- glm(disease/(disease + nondisease) ~ gender + food,  
family = binomial, weights = disease + nondisease, data=babyfood)
```

1.0.1 a) State the model that has been fitted.

The model fitted is a **linear regression model**. This model has been implemented in R via GLM using the binomial family with no link function specified. As no link function has been explicitly called, R will have used the canonical link function for the binomial distribution, which is the logit link function. The weights, w_i , of the linear regression model are total number of infants with and without the disease.

$$g_{\text{canonical}}(\pi_i) = g_{\text{logit}}(\pi_i) = \ln \frac{\pi_i}{(1-\pi_i)}$$

1.0.2 b) Interpret the significance of the parameters for each explanatory variable.

The **girl** parameter for the **gender** explanatory variable is significant at the 5% level, suggesting that gender plays a significant role in whether or not an infant develops respiratory disease in the first year of their lives. The **breast-fed** parameter for the **food** explanatory variable is highly significant, with a p -value of 1.22×10^{-5} meaning the parameter is significant at the 0.1% level.

The **supplement** parameter has a large standard error with respect to its parameter estimate, and thus has a small z -score which is not significant at the 5% or even the 10% level. Therefore the model suggests that taking food supplements has no significant effect on whether an infant develops respiratory disease in the first year of their life.

When dealing with categorical variables in R, one of the classes of the variable is included in the intercept. With us having two categorical variables, *one of the classes from each variable* would have been included in the intercept. For **gender** this would have been the class **boy**, and for **food**

this would have been the class **bottle-fed**. This boy / bottle-fed hybrid baseline parameter is highly significant, with a p -value below 2×10^{-16} , and is therefore clearly significant at the 0.1% significance level. To untangle these parameters from each other to help with our interpretation of the model, we could choose a different class order for each explanatory variable, and thus have different parameters contribute towards the intercept parameter.

1.0.3 c) Does the model provide a good fit?

By looking at the residual deviance of the model in R which represents the model deviance, and knowing that the scale parameter for a binomial GLM, ϕ , is equal to 1, we know that the residual deviance = deviance, $D = \text{scaled deviance}$, S , and thus we can use the residual deviance in a goodness-of-fit test:

$$S = D/\phi = D \sim \chi^2_{n-p}$$

Plugging in the scaled deviance of 0.72192 and the degrees of freedom of $6 - 4 = 2$, we produce the model p -value from the following R code:

```
[52]: deviance <- 0.72192

n <- 6

p <- 4

df.n.sub.p <- n-p

pchisq(deviance, df.n.sub.p, lower.tail = FALSE)
```

0.697006878184402

With a p -value of 0.70 we do not have sufficient evidence to reject the null hypothesis comparing the current model against the saturated model with $n - p$ additional parameters with respect to the current model, and therefore we accept the null hypothesis that the $n - p$ additional parameters in the saturated model are equal to zero, and hence that our model does not require additional parameters. Our model is deemed adequate.

We can therefore say that the model provides a good fit.

It's interesting to look at the explanatory variable contributions to the deviance in the ANOVA table, and it's clear that the **food** variable has the largest and most significant impact of the two variables in reducing the residual deviance.

As we search for an adequate model that's also parsimonious, we may wish to re-fit the model without **gender** and see how the p -value of the model changes.

1.0.4 d) R code to calculate p-value of the model

As shown in c), the code to calculate the p -value for the model:

```
[53]: pchisq(deviance, df.n.sub.p, lower.tail = FALSE)
```

0.697006878184402

1.0.5 e) Interpret the effect of breast-feeding and derive a 95% confidence interval for it

The p-value associated with breast-feeding effect, with an estimated coefficient of -0.6693 and a standard error in that estimate of 0.1530 produces a z statistic of -4.374 , which is highly significant.

One can calculate the p -value (given in the R summary) associated with that z statistic is 1.22×10^{-5} , and thus the breast feeding effect is significant at the 1% level using the following code:

```
[50]: 2*pnorm(-4.374, lower.tail = TRUE)
```

```
1.21990378334922e-05
```

The number of standard errors to produce a 95% confidence interval is given by:

```
[37]: # using the standard normal distribution
num.se.95 <- qnorm(0.025, lower.tail = FALSE)

num.se.95
```

```
1.95996398454005
```

Which is a well-known limit: 1.96 standard errors produces a 95% confidence interval for the standard normal distribution.

We can now plug in these values to produce the 95% confidence interval:

```
[38]: parameter.estimate <- -0.6693
parameter.se <- 0.1530

parameter.estimate + c(-num.se.95*parameter.se,num.se.95*parameter.se)
```

```
1. -0.969174489634628 2. -0.369425510365372
```

Therefore the 95% confidence interval for the breast feeding coefficient is:

$$[-0.969, -0.369]$$

1.0.6 f) Estimate an unbiased estimator of the scale parameter, ϕ .

I.e. what estimator provides: $E[\hat{\phi}] = \phi$?

We can use the property that the expectation of a χ^2 random variable is the number of degrees of freedom of that random variable.

If

$$S = \frac{D}{\phi} \sim \chi_{n-p}^2,$$

then

$$E\left[\frac{D}{\phi}\right] = n - p,$$

and therefore

$$E\left[\frac{D}{n-p}\right] = \phi$$

and thus $\frac{D}{n-p}$ is an unbiased estimator for ϕ :

$$\hat{\phi} = \frac{D}{n-p}$$

```
[54]: scale.estimate <- deviance / df.n.sub.p  
scale.estimate
```

0.36096

$$\hat{\phi} = 0.36096$$

As a sanity check, we'd expect this to be fairly close to 1.

1.0.7 g) Which model would be most reasonable to endorse between:

```
fit2 <- glm(disease/(disease + nondisease) ~ gender + food,  
family = binomial(link="identity"), weights = disease + nondisease)
```

and

```
fit3 <- glm(disease/(disease + nondisease) ~ gender + food,  
family = binomial(link="probit"), weights = disease + nondisease)
```

The only difference between the two models is the choice of link function. I think the **fit3** model with the **probit** link function would be the superior model Vs the **fit2** model which uses the **identity** link function.

Link functions are the mathematical machinery within GLMs to fit non-linear models, mapping the mean value for observation i (i.e. the prediction) to the linear predictor, η_i , via $g(\mu_i) = \eta_i$. The **identity** link function, as the name suggests, provides no non-linearity in the mapping between μ_i and η_i , whereby $\mu_i = \eta_i$, such as is the case in the general linear model.

Since we have seen the non-linear **logit** link function perform well in the above goodness-of-fit tests, we expect that a non-linear S-shaped function, like the **probit** link function, will fit the data better than a linear fit via the **identity** link.

Secondly (but perhaps most importantly), the **probit** link function has a range between 0 and 1 which is essential for fitting a model to a **proportion** with a range between 0 and 1, rather than the linear **identity** function with a range from $-\infty$ to ∞ .

1.0.8 h) For the following model, what is the deviance, D , on d degrees of freedom? And what is d ?

```
fit4 <- glm(disease/(disease + nondisease) ~ gender*food,  
family = binomial, weights = disease + nondisease)
```

Using the **gender*food** rather than **gender+food** syntax, we're adding the following two interaction terms to the model (in addition to the model already fit with 2 degrees of freedom):

- **genderGirl:foodBreast;**
- **genderGirl:foodSuppl.**

Each of the two interaction terms requires a single degree of freedom.

We will therefore have zero degrees of freedom remaining: $d = 0$.

This means that $n - p = 0$ and therefore that p , the number of parameters in the **fit4** model, is equal to the number of observations, n . **fit4** is therefore the saturated model.

This will result in us fitting the saturated model, and thus comparing the saturated model (**fit4**) Vs the saturated model in our hypothesis test, and therefore our generalised LRT, Λ , will equal 1, and therefore our scaled deviance $S = -2 \ln \Lambda$ will equal 0, as $\ln 1 = 0$.

Finally, since the scaled deviance, $S = 0$, then the deviance, $D = 0$.

- $d = 0$
- $D = 0$.

2 Q2)

2.0.1 a) Classification rule derivation

Starting with some definitions:

- π_i is the prior probability that an individual selected at random belongs to population i .
- $C(i|j)$ is the cost of incorrectly allocating an individual to population i , when they really belong to population j .

Bayes rule:

Allocate population 1 if:

$$\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq \frac{\pi_2 C(1|2)}{\pi_1 C(2|1)},$$

otherwise allocate population 2.

In the multivariate normal case (with p variables), where the observation vectors $\mathbf{X}_i \sim \text{MVN}_p(\mu_i, \Sigma)$, then our two probability density functions for population 1 and 2 are:

$$f_1(\mathbf{x}) = \frac{1}{|2\pi\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2}(\mathbf{x} - \mu_1)^T \Sigma^{-1}(\mathbf{x} - \mu_1) \right\},$$

and

$$f_2(\mathbf{x}) = \frac{1}{|2\pi\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2}(\mathbf{x} - \mu_2)^T \Sigma^{-1}(\mathbf{x} - \mu_2) \right\},$$

respectively.

We can therefore calculate the likelihood ratio as:

$$\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} = \exp \left\{ -\frac{1}{2}(\mathbf{x} - \mu_1)^T \Sigma^{-1}(\mathbf{x} - \mu_1) + \frac{1}{2}(\mathbf{x} - \mu_2)^T \Sigma^{-1}(\mathbf{x} - \mu_2) \right\},$$

where the $\frac{1}{|2\pi\Sigma|^{1/2}}$ terms in each pdf cancel out.

We can then calculate the log-likelihood by taking natural logs of both sides:

$$\ln \left(\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \right) = -\frac{1}{2}(\mathbf{x} - \mu_1)^T \Sigma^{-1}(\mathbf{x} - \mu_1) + \frac{1}{2}(\mathbf{x} - \mu_2)^T \Sigma^{-1}(\mathbf{x} - \mu_2)$$

and factorising the $\frac{1}{2}$:

$$\ln \left(\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \right) = -\frac{1}{2} \left((\mathbf{x} - \mu_1)^T \Sigma^{-1}(\mathbf{x} - \mu_1) - (\mathbf{x} - \mu_2)^T \Sigma^{-1}(\mathbf{x} - \mu_2) \right).$$

We can expand out the $(\mathbf{x} - \mu_1)^T \Sigma^{-1}(\mathbf{x} - \mu_1) - (\mathbf{x} - \mu_2)^T \Sigma^{-1}(\mathbf{x} - \mu_2)$ terms:

Firstly the $(\mathbf{x} - \mu_1)^T \Sigma^{-1}(\mathbf{x} - \mu_1)$ term:

$$\begin{aligned} &= (\mathbf{x}^T \Sigma^{-1} - \mu_1^T \Sigma^{-1})(\mathbf{x} - \mu_1) \\ &= \mathbf{x}^T \Sigma^{-1} \mathbf{x} - \mathbf{x}^T \Sigma^{-1} \mu_1 - \mu_1^T \Sigma^{-1} \mathbf{x} + \mu_1^T \Sigma^{-1} \mu_1 \end{aligned}$$

Each of these four components is a scalar, and the transpose of a scalar is the scalar itself, i.e. $a^T = a$, therefore:

$$\mu_1^T \Sigma^{-1} \mathbf{x} = (\mu_1^T \Sigma^{-1} \mathbf{x})^T = \mathbf{x}^T (\Sigma^{-1})^T \mu_1 = \mathbf{x}^T (\Sigma^T)^{-1} \mu_1 = \mathbf{x}^T \Sigma^{-1} \mu_1,$$

since Σ is a symmetric matrix, $\Sigma = \Sigma^T$.

We can therefore write:

$$(\mathbf{x} - \mu_1)^T \Sigma^{-1}(\mathbf{x} - \mu_1) = \mathbf{x}^T \Sigma^{-1} \mathbf{x} - 2\mathbf{x}^T \Sigma^{-1} \mu_1 + \mu_1^T \Sigma^{-1} \mu_1$$

and

$$(\mathbf{x} - \mu_2)^T \Sigma^{-1}(\mathbf{x} - \mu_2) = \mathbf{x}^T \Sigma^{-1} \mathbf{x} - 2\mathbf{x}^T \Sigma^{-1} \mu_2 + \mu_2^T \Sigma^{-1} \mu_2.$$

Thus we can expand out:

$$(\mathbf{x} - \mu_1)^T \Sigma^{-1}(\mathbf{x} - \mu_1) - (\mathbf{x} - \mu_2)^T \Sigma^{-1}(\mathbf{x} - \mu_2)$$

to

$$-2\mathbf{x}^T \Sigma^{-1} \mu_1 + \mu_1^T \Sigma^{-1} \mu_1 + 2\mathbf{x}^T \Sigma^{-1} \mu_2 - \mu_2^T \Sigma^{-1} \mu_2,$$

where we can factor out the $-2\mathbf{x}^T \Sigma^{-1}$ terms:

$$-2\mathbf{x}^T \Sigma^{-1}(\mu_1 - \mu_2) + \mu_1^T \Sigma^{-1} \mu_1 - \mu_2^T \Sigma^{-1} \mu_2.$$

The $\mu_1^T \Sigma^{-1} \mu_1 - \mu_2^T \Sigma^{-1} \mu_2$ terms follow the difference of two squares, and can be written as $(\mu_1 + \mu_2)^T \Sigma^{-1}(\mu_1 - \mu_2)$.

Our log-likelihood can therefore be written:

$$\begin{aligned} \ln \left(\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \right) &= -\frac{1}{2} \left(-2\mathbf{x}^T \Sigma^{-1}(\mu_1 - \mu_2) + (\mu_1 + \mu_2)^T \Sigma^{-1}(\mu_1 - \mu_2) \right) \\ &= \mathbf{x}^T \Sigma^{-1}(\mu_1 - \mu_2) - \frac{1}{2}(\mu_1 + \mu_2)^T \Sigma^{-1}(\mu_1 - \mu_2) \end{aligned}$$

The common factor in both terms is $\Sigma^{-1}(\mu_1 - \mu_2)$. Let $\mathbf{L} = \Sigma^{-1}(\mu_1 - \mu_2)$ such that our log-likelihood takes the form:

$$\ln \left(\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \right) = \mathbf{x}^T \mathbf{L} - \frac{1}{2}(\mu_1 + \mu_2)^T \mathbf{L}.$$

Note again that each of the two above terms is a scalar (\mathbf{L} is a $p \times 1$ vector), so we can shuffle the order of the terms by taking the transpose of each:

$$\ln \left(\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \right) = \mathbf{L}^T \mathbf{x} - \frac{1}{2} \mathbf{L}^T (\mu_1 + \mu_2).$$

Recall that as:

$$\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq \frac{\pi_2 C(1|2)}{\pi_1 C(2|1)},$$

the log-likelihood relationship must be:

$$\ln \left(\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \right) \geq \ln \left(\frac{\pi_2 C(1|2)}{\pi_1 C(2|1)} \right),$$

therefore:

$$\mathbf{L}^T \mathbf{x} - \frac{1}{2} \mathbf{L}^T (\mu_1 + \mu_2) \geq \ln \left(\frac{\pi_2 C(1|2)}{\pi_1 C(2|1)} \right).$$

Our allocation rule is therefore:

Allocate to population 1 if:

$$\mathbf{L}^T \mathbf{x} - \frac{1}{2} \mathbf{L}^T (\mu_1 + \mu_2) \geq \ln \left(\frac{\pi_2 C(1|2)}{\pi_1 C(2|1)} \right),$$

otherwise allocate to population 2.

b) Mahalanobis distance and misclassification probability.

We are provided with the mean results for four tests administered to two groups.

We are also provided with the pooled *within-groups* sample covariance matrix, \mathbf{S}_U , which is an unbiased estimator for the population covariance matrix, Σ , whereby:

$$\mathbf{S}_U = \frac{(n_1 - 1)\mathbf{S}_{1U} + (n_2 - 1)\mathbf{S}_{2U}}{n_1 + n_2 - 2}.$$

The squared Mahalanobis distance (a measure of distance between two **population** means), α , is calculated as:

$$\alpha = (\mu_1 - \mu_2)^T \Sigma^{-1} (\mu_1 - \mu_2).$$

Since we don't know the population parameters, we can calculate the sample squared Mahalanobis distance:

$$\hat{\alpha} = (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T \mathbf{S}_U^{-1} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2).$$

We therefore take the inverse of the pooled sample covariance matrix, \mathbf{S}_U^{-1} , which is provided as **invSp** in the question, and calculate $\hat{\alpha}$.

```
[69]: # sample mean vector for senile group
senile.37 <- c(12.57, 9.57, 11.49, 7.97)

# sample mean vector for non-senile group
non.senile.12 <- c(8.75, 5.35, 8.50, 4.75)

# the four different tests performed by each individual
subtest <- c('Information', 'Similarities', 'Arithmetic', 'Picture completion')

df.means <- data.frame(subtest, senile.37, non.senile.12)

df.means
```

	subtest <fct>	senile.37 <dbl>	non.senile.12 <dbl>
A data.frame: 4 × 3	Information	12.57	8.75
	Similarities	9.57	5.35
	Arithmetic	11.49	8.50
	Picture completion	7.97	4.75

```
[70]: # symmetric sample covariance matrix
sample.covar <- data.frame(Information = c(11.2553, 9.4042, 7.1489, 3.3830),
  Similarities = c(9.4042, 13.5318, 7.3830, 2.5532),
  Arithmetic = c(7.1489, 7.3830, 11.5744, 2.6170),
  PictureCompletion = c(3.3830, 2.5532, 2.6170, 5.8085))

sample.covar.inv <- solve(sample.covar)

sample.covar.inv
```

	Information	Similarities	Arithmetic	PictureCompletion
A matrix: 4 × 4 of type dbl	0.25907356	-0.13576450	-0.05877998	-0.06473009
	-0.13576450	0.18645117	-0.03833003	0.01438476
	-0.05877998	-0.03833003	0.15098314	-0.01694172
	-0.06473009	0.01438476	-0.01694172	0.21117177

2.0.2 Squared Mahalanobis distance calculation:

```
[79]: alpha.hat <- t(senile.37 - non.senile.12) %*% sample.covar.inv %*% (senile.37 -
  ↪ non.senile.12)

alpha.hat[1,1]
```

2.42533274943634

We estimate the squared Mahalanobis distance to be 2.43.

The probability that an individual is misallocated to the correct group, i.e. is misclassified, is given by the standard normal distribution function $= \Phi\left(-\frac{\sqrt{\hat{\alpha}}}{2}\right)$

```
[77]: pnorm(-0.5 * sqrt(alpha.hat[1,1]), lower.tail = TRUE)
```

0.218085889888665

We calculate a misclassification probability of 0.22.

Note, this probability may be *underestimated* for small sample sizes.

2.0.3 (c) PCA

100 individuals were given 3 scores based on a series of tests designed to measure an underlying attitude.

- (i) The third eigenvector has an eigenvalue of zero. This means there's an exact linear relationship between the three variables (the third eigenvector not being orthogonal to the other two). The third principal component accounts for *zero variance* with respect to the other two, and therefore that one of the variables is redundant.
- (ii) The first principal component accounts for $\frac{296.724}{296.724 + 25.276} = 92\%$ of the variance. This dominating component is a measure of the overall size, with all three coefficients having similar magnitude and *sign*.

The second principal component accounts for the remaining 8% of the variance (as the third component accounts for zero variance), but interestingly, shows there to be a *contrast between the second and third variables*, with opposite sign, whilst the first variable has no impact.

There's therefore interest in the interplay between the second and third variables, and since one of the variables should be removed, it makes sense that it's the first.

When re-running the PCA on the second and third variable, you'd be running PCA on the following unbiased sample covariance matrix:

$$\mathbf{S}_U = \begin{bmatrix} 52 & 36 \\ 36 & 73 \end{bmatrix}$$

Where the expression to find the eigenvalues, indicating the proportions of the variance accounted for by each of the two principal components is:

$$\det \begin{bmatrix} 52 - \lambda & 36 \\ 36 & 73 - \lambda \end{bmatrix} = 0$$

I'd expect there to still be an overall size component as the first principal component that accounts for most of the variance, where each coefficient has the same sign. I'd then expect the second component, with a significantly lower eigenvalue / variance, to contain coefficients with the **same coefficient values, but of opposite sign**.

This is because we're down to two variables and two principal components, where each component is a linear combination of the two variables, the size of the weights for each variable in the two linear combinations won't change. But, the sign will in the second principal component, as we've

seen there to be a contrasting interaction between the second and third variables when performing PCA initially, with the three variables.

This initial variable removal process of finding and removing highly correlated variables is useful as a pre-processing step prior to fitting a regression. Highly correlated variables can cause enormous errors in regression coefficients, which is clearly sub-optimal.

Note that by reducing the number of variables, we've reduced the dimensionality of our data, from three dimensions to two, by *eliminating* one of our underlying variables. This is one of the two ways in which PCA can be used to reduce the dimensions in our data. The other way, is to select the principal components that explain most of the variance in the data, where these components represent linear combinations of the underlying variables, where we can treat these principal components as 'new' variables, and hence work with fewer number of principal components than our original number of variables, whilst maintaining most of the variance in the data in this new representation.

As we've also reduced the number of our variable dimensions from 3 to 2, we can more easily visually interpret the interplay between the two variables in a plot (I find 3-D plots often tricky to interpret!), as the PCA analysis has revealed the best 2-D projection of the test score data.

This iterative PCA process could also help to feedback to the test designers that the three scores that they've produced aren't truly orthogonal when it comes to measuring the underlying attitude - essentially one of those tests has been wasted.

As an interesting aside, I suspect the PCA analysis from the above would look similar to the PCA analysis that would result from comparing the returns of two similar stocks in a paired analysis: e.g. Coke and Pepsi. The results of which are shown below:

```
[129]: # dataframe with returns for Coca Cola and Pepsi going back to the 1960s.
df.pair <- read.csv('coke_pepsi_paired_returns.csv')

df.pair[1:5,]
```

	return_coke <dbl>	return_pepsi <dbl>
	-0.003780718	-0.002890173
A data.frame: 5 × 2	-0.001897533	-0.014492754
	-0.008593156	-0.008823529
	-0.004755695	-0.005934718
	0.001233141	0.000000000

```
[125]: # producing the sample covariance matrix
S <- var(df.pair)
S
```

		return_coke	return_pepsi
A matrix: 2 × 2 of type dbl	return_coke	0.0002328181	0.0001219119
	return_pepsi	0.0001219119	0.0002464368

```
[127]: # producing the eigenvalues and eigenvectors
eig <- eigen(S)
eig
```

```
eigen() decomposition
$values
[1] 0.0003617293 0.0001175255

$vectors
      [,1]      [,2]
[1,] 0.6871070 -0.7265563
[2,] 0.7265563  0.6871070
```

```
[128]: # proportion of variance between the first and second principal components
(eig$values/sum(eig$values)) * 100
```

```
1. 75.4774469270531 2. 24.5225530729469
```

END OF ASSIGNMENT
