

FIT5197

DATA MODELLING

ASSIGNMENT-2

NIKHIL TARALKAR

28980433

Task A: Linear Regression

Build a linear regression model using the specific “auto mpg train.csv” provided with the assignment to predict mpg (mile per gallon). The second file “auto mpg test.csv” will be used for evaluation.

Tasks A.1:

There are some missing values listed as “?”. Describe your strategy for treating missing values and update (edit by hand) the file accordingly.

- There are around 6 missing values with “?” in the auto mpg train dataset in horsepower column. In order to handle these missing values, I have replaced them manually with mean of the column within the excel file itself. The mean of column is 105.30

28	4	140	90	2264	15.5	71	1	chevrolet vega 2300
25	4	113	95	2228	14	71	3	toyota corona
25	4	98	?	2046	19	71	1	ford pinto
19	6	232	100	2634	13	71	1	amc gremlin
16	6	225	105	3439	15.5	71	1	plymouth satellite custom
17	6	250	100	3329	15.5	71	1	chevrolet chevelle malibu
19	6	250	88	3302	15.5	71	1	ford torino 500

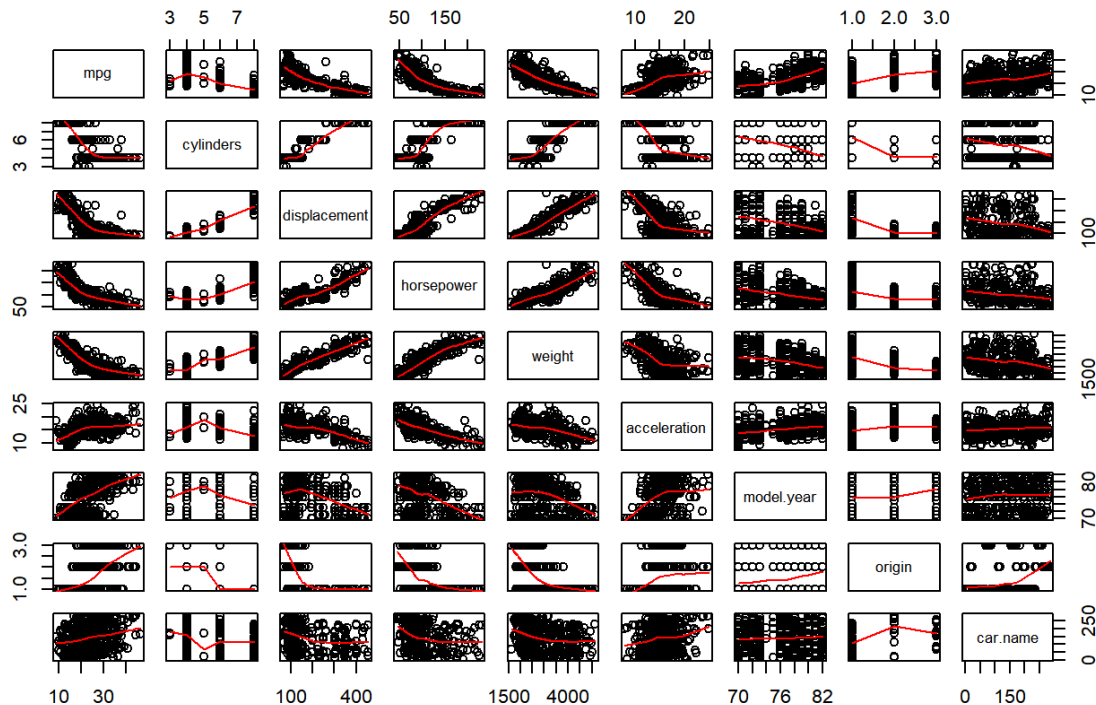
Replaced with 105.30 (mean value)

28	4	140	90	2264	15.5	71	1	chevrolet vega 2300
25	4	113	95	2228	14	71	3	toyota corona
25	4	98	105.3	2046	19	71	1	ford pinto
19	6	232	100	2634	13	71	1	amc gremlin
16	6	225	105	3439	15.5	71	1	plymouth satellite custom
17	6	250	100	3329	15.5	71	1	chevrolet chevelle malibu
19	6	250	88	3302	15.5	71	1	ford torino 500

Task A.2:

Pair plot mpg vs. the other variables to visualize the relationships and discuss what you see.

Figure 1: Pairs Plot for MPG



Task A.3:

Based on your pair plots, propose an initial set of variables to use for a multiple linear regression model to predict mpg.

```
> cor(at[,1:8])
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model.year	origin
mpg	1.0000000	-0.7717897	-0.7991855	-0.7852616	-0.8306991	0.4581737	0.5981966	0.5465047
cylinders	-0.7717897	1.0000000	0.9502605	0.8393205	0.8959469	-0.5277690	-0.3739750	-0.5494493
displacement	-0.7991855	0.9502605	1.0000000	0.8975662	0.9327988	-0.5746634	-0.3981105	-0.5962623
horsepower	-0.7852616	0.8393205	0.8975662	1.0000000	0.8678581	-0.6987341	-0.4446407	-0.4516817
weight	-0.8306991	0.8959469	0.9327988	0.8678581	1.0000000	-0.4534332	-0.3317539	-0.5665474
acceleration	0.4581737	-0.5277690	-0.5746634	-0.6987341	-0.4534332	1.0000000	0.3186810	0.2224061
model.year	0.5981966	-0.3739750	-0.3981105	-0.4446407	-0.3317539	0.3186810	1.0000000	0.1955691
origin	0.5465047	-0.5494493	-0.5962623	-0.4516817	-0.5665474	0.2224061	0.1955691	1.0000000

Based on the correlation between the columns of the auto mpg train dataset, I have chosen the columns which shows high correlation. For e.g. displacement, horsepower, weight, cylinders are highly correlated to mpg. Negative correlation shows indirect relationship i.e. one value increases while other decreases and hence negative.

Task A.4:

With variables of your choice build the model using the `lm()` routine in R, and then print the summary of the model to get the R diagnostics. Briefly explain the statistics in the summary, e.g. R² value, t-value, standard error, p-value (ignoring the F-statistics line). What does this imply about the predictors for your model?

```
model <- lm(formula = mpg ~ cylinders + weight + displacement + horsepower, data=mpg_data)
summary(model)
```

```
Call:
lm(formula = mpg ~ cylinders + weight + displacement + horsepower,
    data = mpg_data)
```

```
Residuals:
    Min       1Q   Median       3Q      Max
-11.7388  -2.8605  -0.4091   2.4691  15.8292
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  47.4852274   1.6675139   28.477  < 2e-16 ***
cylinders    -0.4971540   0.4406348   -1.128  0.259995
weight      -0.0055864   0.0007913   -7.060  9.31e-12 ***
displacement  0.0065369   0.0097270    0.672  0.502015
horsepower  -0.0538553   0.0138177   -3.898  0.000117 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 4.349 on 343 degrees of freedom
Multiple R-squared:  0.7079,    Adjusted R-squared:  0.7045
F-statistic: 207.8 on 4 and 343 DF,  p-value: < 2.2e-16
```

Residual standard error: 4.349 on 343 degrees of freedom

Multiple R-squared: 0.7079, Adjusted R-squared: 0.7045

F-statistic: 207.8 on 4 and 343 DF, p-value: < 2.2e-16

R-square and most common metric to judge the performance of regression models. R² measures.

R-squared is always between 0 and 1:

- 0 indicates that the model explains NIL variability in the response data around its mean.
- 1 indicates that the model explains full variability in the response data around its mean.

In our case, R-square value is 0.709 which can be improved with different predictors and show below.

Adjusted R-squared is nothing but the change of R-square that adjusts the number of terms in a model. Adjusted R square calculates the proportion of the variation in the dependent variable accounted by the explanatory variables.

The **p-value** basically tests the null hypothesis for each term that the coefficient is equal to zero (no effect). A low p-value (< 0.05) shows that you can reject the null hypothesis. Consequently, it can be said that a low p-value is a addition to your model because changes in the predictors value is related to changes in the response. In contrast, a larger p-value shows that changes in the predictor are not associated with the changes int the response. Horsepower is supporting null hypothesis since its value is close to 0 which is 0.000117.

In the output above, we can see that the predictor variables of horsepower and weight are significant because both of their p-values are less than 0.05. However, the p-value for displacement (0.50) and cylinders (0.25) is greater than the common alpha level of 0.05, which indicates that it is not statistically significant

The **t-value** measures the size of the difference relative to the variation in your sample data.

In other words, T value is simply the calculated difference represented in units of standard error. The greater the magnitude of T (it can be either positive or negative), the greater the evidence *against* the null hypothesis that there is no significant difference. The closer T is to 0, the more likely there isn't a significant difference.

Standard error represents the average distance that the observed values fall from the regression line. Conveniently, it tells you how wrong the regression model is on average using the units of the response variable.

The standard error of this regression coefficient captures how much uncertainty is associated with this coefficient.

A model is said to have best fit with minimum standard error. In this case, all our predictors have low standard errors. This indicates that the model is having a good fit. The standard error for the weight is the lowest i.e. 0.00558, followed by displacement and horsepower, which is 0.062 and 0.5331.

Residual shows the difference between observed and predicted values. The quartiles show that our residuals are around zero or not. With above result i.e. Median is -0.4 which is very close to 0 and 1Q and 3Q are -2.86 and 2.46 on the opposite side of the Y axis. So, we can say that our residuals are distributed in a shape of very close to normal distribution.

Task A.5:

Test the fitted model using the “**auto mpg test.csv**”, and calculate the MSE on the test set, reporting it. Note the test set has no missing values.

- For question A.5, I have used Metrics library to calculate MSE on the test set. MSE can be directly calculated by passing on the predicted and test variable values as shown in the below example. MSE for cylinders, weight, displacement and horsepower predictors is 14.75 which is very high and can be improved and done in the next task.

```
#importing library
library(Metrics)
```

```
model <- lm(formula = mpg ~ cylinders + weight + displacement + horsepower, data=mpg_data)
```

```
# calculating MSE value
mse(model_prediction, auto_data$mpg)
```

MSE: 14.75972

Task A.6:

Can you improve your model with different predictors? Try out some different ratios or products of the better predictor variables. How will you evaluate the different alternative predictors on your existing model (not using the test set)? Evaluate them and suggest which single predictor you would like to add (or none, if it looks like none would improve it). If you suggest adding a single predictor, then add it and repeat step A.5 to evaluate it on the test set.

Yes, I can improve the model by including different predictors such as model year and origin.

Model 1:

```
model1 <- lm(formula = mpg ~ cylinders + weight + displacement + horsepower + model.year, data=mpg_data)
```

Multiple R-squared: 0.809, Adjusted R-squared: 0.8062

MSE: 8.662696

As you can the MSE has been improved by including model.year as predictor to 8.66 which can still be improved further.

Model 2:

```
model2 <- lm(formula = mpg ~ cylinders + origin + displacement + weight + model.year, data=mpg_data)
```

Multiple R-squared: 0.8177, Adjusted R-squared: 0.815

MSE: 7.207671

Again, the model is improved by adding origin and removing horsepower predictors from the model.

Model 3:

```
model3 <- lm(formula = mpg ~ origin + weight + model.year, data=mpg_data)
summary(model3)
```

Multiple R-squared: 0.8149, Adjusted R-squared: 0.8133

MSE: 6.530443

The model is said to be fit with minimum MSE and hence our target is to bring minimum MSE and bring the best fit. From the Model 3, we can conclude that origin, weight and model year are the best predictors for mpg with minimum MSE of 6.53 and Maximum R-square of 0.814.

Task B: Logistic Regression

Build a logistic regression model using the specific “adult income train.csv” provided with the Assignment to predict the income variable. The second file “adult income test.csv” will be used for evaluation.

Tasks B.1:

There are some missing values listed as “?”. Describe your strategy for treating missing values, but note sometimes it is OK to leave missing value as a separate categorical value (we call this “missing informative”). Note there are too many to edit by hand, so if you wish to modify them, identify them with a Boolean test like `data$workclass[id]=='?'` and modify the values in a loop.

From the dataset, we can see that there are plenty of missing values with “?”. These missing values are replaced by a new category “**Non-specified**” defined manually. Knowing that values can impact my prediction, I didn’t drop the index. Instead I have created new category or factor level that helps to make my prediction more precise. Below diagram shows “?” replaced with Non-specified in the table.

age	workclass	fnlwgt	education	educational_num	marital_status	occupation	relationship	race	gender
18	Non-specified	103497	Some-college	10	Never-married	Non-specified	Own-child	White	Female
29	Non-specified	227026	HS-grad	9	Never-married	Non-specified	Unmarried	Black	Male
58	Non-specified	299831	HS-grad	9	Married-civ-spouse	Non-specified	Husband	White	Male
40	Private	85019	Doctorate	16	Married-civ-spouse	Prof-specialty	Husband	Asian-Pac-Islander	Male
72	Non-specified	132015	7th-8th	4	Divorced	Non-specified	Not-in-family	White	Female
65	Non-specified	191846	HS-grad	9	Married-civ-spouse	Non-specified	Husband	White	Male
41	Private	109912	Bachelors	13	Never-married	Other-service	Not-in-family	White	Female
17	Non-specified	165361	10th	6	Never-married	Non-specified	Own-child	White	Male
44	Self-emp-inc	223881	HS-grad	9	Married-civ-spouse	Craft-repair	Husband	White	Male
41	Non-specified	38434	Masters	14	Married-civ-spouse	Non-specified	Wife	White	Female

Task B.2:

With all variables, build a model using the `glm (income~.,family=binomial, data=???)` routine in R, and then print the summary of the model to get the R diagnostics. Briefly explain the statistics in the summary, e.g. Z-value, standard error, p-value. What does this imply about the predictors for your model? Notice many of the variables are multi-valued categorical, and in most cases only some of the values are significant.

The main objective in this task is to predict income using logistic regression model. Below R diagnostics shows the complete summary of the logistic regression model.

Call:

```
glm(formula = income ~ ., family = binomial, data = income_data_train)
```

Deviance Residuals:

```
      Min       1Q   Median       3Q      Max
-5.1131  -0.5027  -0.1823  -0.0336   3.8667
```

Coefficients: (2 not defined because of singularities)

```
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -6.700e+00  6.587e-01 -10.171  < 2e-16 ***
age          2.493e-02  1.421e-03  17.547  < 2e-16 ***
workclassLocal-gov -6.918e-01  9.684e-02  -7.144  9.06e-13 ***
workclassNever-worked -9.352e+00  8.524e+01  -0.110  0.912637
workclassNon-specified -1.294e+00  1.191e-01 -10.858  < 2e-16 ***
workclassPrivate -5.237e-01  8.067e-02  -6.491  8.50e-11 ***
```

workclassSelf-emp-inc	-3.686e-01	1.061e-01	-3.474	0.000513	***
workclassSelf-emp-not-inc	-1.063e+00	9.446e-02	-11.257	< 2e-16	***
workclassState-gov	-8.881e-01	1.081e-01	-8.213	< 2e-16	***
workclasswithout-pay	-1.402e+00	7.893e-01	-1.776	0.075699	.
fnlwgt	7.803e-07	1.473e-07	5.298	1.17e-07	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 48173 on 43841 degrees of freedom
 Residual deviance: 27615 on 43743 degrees of freedom
 AIC: 27813

Number of Fisher Scoring iterations: 11

The “Pr(>[z])” is also called as “**p-value**” which indicates whether the coefficient point estimate is different from 0. Any values which is lower than 0.05 is considered as good for the model. **p-value** helps you to decide whether there is a relationship between two variables or not. The smaller the p-value this mean the more confident you are about the existence of relationship between the two variables. In this case, values ending with *** are more significant values while values without * or . are non-significant values.

The **Std.error** is the Standard deviation of the coefficient point estimate in the GLM model. It is a measure of the uncertainty about this estimate – if the value is too large, then the coefficient point estimate is calculated with a lot of imprecision. However, if the Std.error value is low, it is good for the prediction.

z-value: Z-value is the ratio of estimate value and Std.error. It is the regression coefficient divided by its standard error.

Null Deviance and Residual Deviance – Null Deviance indicates the response predicted by a model with nothing but an intercept. Lower the value, better the model. Residual deviance indicates the response predicted by a model on adding independent variables. Lower the value, better the model.

AIC (Akaike Information Criteria) – The analogous metric of adjusted R^2 in logistic regression is AIC. AIC is the measure of fit which penalizes model for the number of model coefficients. Therefore, we always prefer model with minimum AIC value.

Task B.3:

Test the fitted model using the “adult income test.csv”, and calculate the confusion matrix on the test set, reporting it. Also, give the precision, accuracy and recall (Lecture 3). Note the test set has no missing values.

The task was to predict income based on the adult income test dataset using logistic regression model. Below given confusion matrix gives the exact summary of the statistics along with precision and recall achieved in predicting income values.

```
> confusionMatrix(income_test_data$income, prediction_new)
Confusion Matrix and Statistics
```

	Reference	
Prediction	0	1
0	3507	264
1	493	736

Accuracy : 0.8486


```

          95% CI : (0.8384, 0.8584)
No Information Rate : 0.8
P-Value [Acc > NIR] : < 2.2e-16

          Kappa : 0.5643
McNemar's Test P-Value : < 2.2e-16

          Sensitivity : 0.8768
          Specificity : 0.7360
          Pos Pred Value : 0.9300
          Neg Pred Value : 0.5989
          Prevalence : 0.8000
          Detection Rate : 0.7014
          Detection Prevalence : 0.7542
          Balanced Accuracy : 0.8064

          'Positive' Class : 0

```

Accuracy:

```

> # finding Accuracy
> N <- nrow(income_test_data) #Number of observations
> diag <- diag(conf) #TN and TP
> Accuracy <- sum(diag)/N #Accuracy = (TP + TN)/N
> round(Accuracy*100,2)
[1] 84.86

```

Precision and Recall:

Precision <dbl>	Recall <dbl>	F1 <dbl>	Actual.Dist <dbl>	Predicted.Dist <dbl>
87.67	93.00	90.26	75.42	80
73.60	59.89	66.04	24.58	20

From the above table we can see that Precision and Recall values are larger for type 0. Also, the Predicted.Dist is larger than Actual.Dist for type=0, therefore, we can conclude that the model is more biased towards the class 0.

Task B.4:

Can you improve your model with different predictors? For instance, you might reconstruct the categorical features to only include significant values and then have an “other” value that groups together all non-significant ones. Perhaps the best way to do this is to create a new data frame with your modified attributes and build the model on that using the R construct “income~.” Report the R diagnostics and the confusion matrix and other scores on the test set (as per B.3) for the new model and comment on the difference

Yes, the model can be improved with different predictors, by only considering the most significant values. From the previous model used in task 3, I have only considered the most significant values and categorized non-significant values into Others. The z-values which are represented with * significant ones while the one which are with (. And with *) are non-significant values. I have only considered significant values for my analysis. For e.g.

Coefficients: (2 not defined because of singularities)

	Estimate	Std. Error	z value	Pr(> z)	
(Intercept)	-6.700e+00	6.587e-01	-10.171	< 2e-16	***
age	2.493e-02	1.421e-03	17.547	< 2e-16	***
workclassLocal-gov	-6.918e-01	9.684e-02	-7.144	9.06e-13	***
workclassNever-worked	-9.352e+00	8.524e+01	-0.110	0.912637	
workclassNon-specified	-1.294e+00	1.191e-01	-10.858	< 2e-16	***
workclassPrivate	-5.237e-01	8.067e-02	-6.491	8.50e-11	***
workclassSelf-emp-inc	-3.686e-01	1.061e-01	-3.474	0.000513	***
workclassSelf-emp-not-inc	-1.063e+00	9.446e-02	-11.257	< 2e-16	***
workclassState-gov	-8.881e-01	1.081e-01	-8.213	< 2e-16	***
workclassWithout-pay	-1.402e+00	7.893e-01	-1.776	0.075699	.
fnlwgt	7.803e-07	1.473e-07	5.298	1.17e-07	***

For workclass category, only "Never-worked" and "Without-pay" are non-significant values and therefore categorized into "Others". Similarly, done for rest of the columns.

To do this, I have taken a new data frame which is a copy of training dataset. Before proceeding further, I have changed the data type of the columns. since, the grouping does not work on the factor levels, I have converted the columns back to their original datatype. For e.g. workclass was changed to character datatype and then grouped into others category for "Never-worked" and "withpout-pay" values which are non-significant. Similar logic is applied for rest of the columns.

Then, the model was build on new data frame of training dataset. This model was then used to predict values for income for the test dataset.

Confusion Matrix:

```
> confusionMatrix(income_test_data$income, prediction_new_value)
Confusion Matrix and Statistics
```

	Reference	
Prediction	0	1
0	3505	266
1	494	735

Accuracy : 0.848
95% CI : (0.8377, 0.8578)
No Information Rate : 0.7998
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.5627
McNemar's Test P-Value : < 2.2e-16

Sensitivity : 0.8765
Specificity : 0.7343
Pos Pred Value : 0.9295
Neg Pred Value : 0.5980
Prevalence : 0.7998
Detection Rate : 0.7010
Detection Prevalence : 0.7542
Balanced Accuracy : 0.8054

'Positive' Class : 0

Accuracy:

```

> N <- nrow(income_test_data)
> diag <- diag(conf_matrix)
> Accuracy <- sum(diag)/N
> round(Accuracy*100,2)
[1] 84.8
>

```

Precision and Recall:

Precision <dbl>	Recall <dbl>	F1 <dbl>	Actual.Dist <dbl>	Predicted.Dist <dbl>
87.65	92.95	90.22	75.42	79.98
73.43	59.80	65.92	24.58	20.02

From the above table, we can see that Precision and recall values are larger for type 0. Also, the Predicted.Dist is larger than Actual.Dist for type 0. Therefore, we can conclude that the model is more biased towards the class 0.

Precision - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. High precision relates to the low false positive rate. We have got 0.876 precision which is pretty good.

Precision = $TP / (TP + FP)$

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class. We have got recall of 0.631 which is good for this model as it's above 0.5.

Recall = $TP / (TP + FN)$

From the above two model, we can say that there is not much improvement in the model. As the accuracy is increased from 84.86 to 84.88. Therefore, it can be concluded grouping significant values does not make any difference in this case.

Task C: Sampling

C1. First write a sampling algorithm that uses the rejection method (for either one of the versions $p(x|\lambda)$ above). In your report describe how this was designed. Define this in R, sample 1000 values and histogram them.

$$p(x|\lambda) = \frac{1}{1 - e^{-2\lambda}} \lambda e^{-\lambda x} \quad \text{for } x \in [0, 2]$$

From the above Probability distribution function, samples need to be generated. However, to implement sampling rejection method, we need Y limit. To reject all the samples that lie with the area of probability distribution function. For this I have calculated values of y at each value of x at 0.01 intervals. From this, the Y-limit obtained is 1.578.

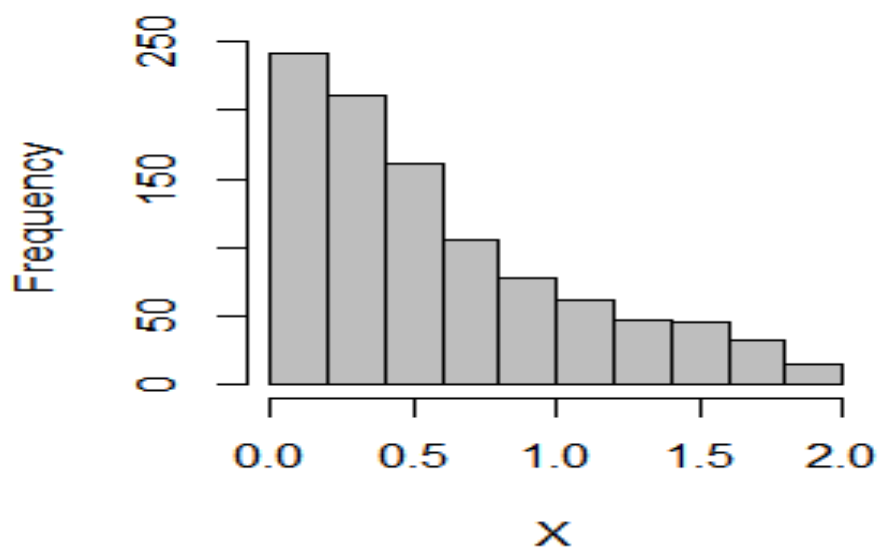
```
#function to obtain y-value
df_1 <- data.frame()
for(x in seq(from=0, to=2, by=0.01))
{
  lambda = 1.5
  px <- (lambda * exp(-(lambda) * x)) / (1 - exp(-(2) * lambda))

  df <- data.frame(px)
  df_1 <- rbind(df_1, df)
}

print(max(df_1))
```

The main function Sample_function1 then calculates the values for probability distribution for x values between 0 and 2. If the y value is less than the probability values than x is returned from the sample_function1. Like this, the rejection sampling method will remove all the values that are greater than y . After that plotting histogram for 1000 samples.

Rejection sampling of 1000 values

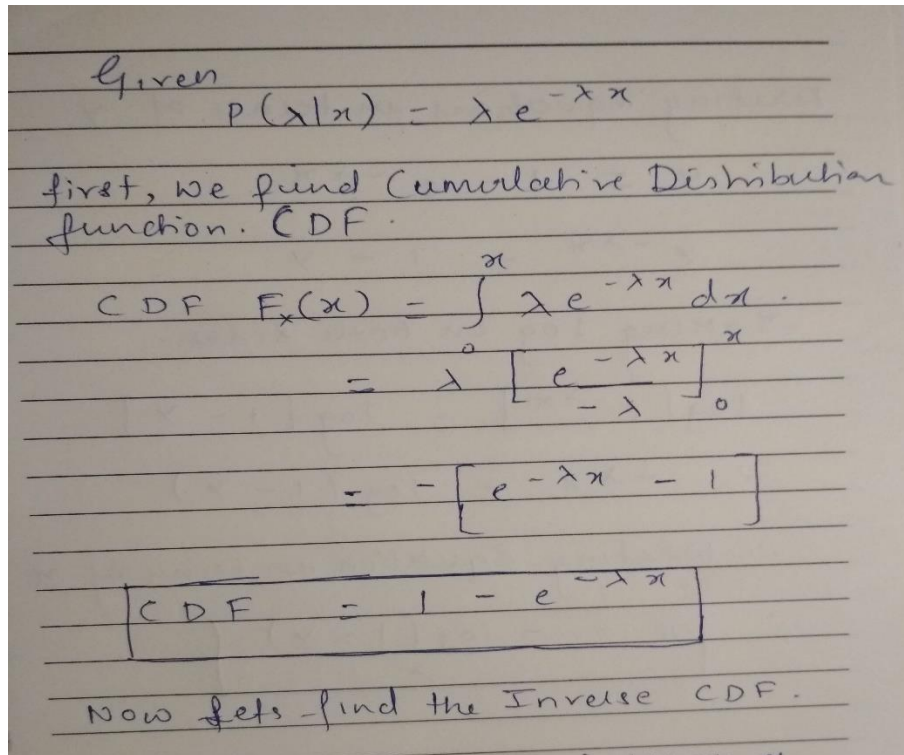


Task C.2:

C2. Second write a sampling algorithm that uses the inverse sampling method (for either one of the versions $p(x|\lambda)$ above). In your report describe how this was designed. Define this in R, sample 1000 values and histogram them.

$$p(x|\lambda) = \lambda e^{-\lambda x}$$

For inverse sampling method, I have used above probability distribution function. Now to implement inverse sampling method, first we need to obtain inverse of above given distribution function. Below two snippets give a steps done to convolute inverse CDF function.



Handwritten derivation of the Cumulative Distribution Function (CDF) for an exponential distribution with parameter λ .

Given $p(x|\lambda) = \lambda e^{-\lambda x}$

first, we find Cumulative Distribution function. CDF.

$$\begin{aligned} \text{CDF } F_x(x) &= \int_0^x \lambda e^{-\lambda x} dx \\ &= \lambda \left[\frac{e^{-\lambda x}}{-\lambda} \right]_0^x \\ &= - \left[e^{-\lambda x} - 1 \right] \\ \boxed{\text{CDF} &= 1 - e^{-\lambda x}} \end{aligned}$$

Now lets find the Inverse CDF.

Deriving equations in terms of y

$$y = 1 - e^{-\lambda x}$$

$$e^{-\lambda x} = 1 - y$$

Taking log on both sides.

$$\therefore \log[e^{-\lambda x}] = \log[1 - y]$$

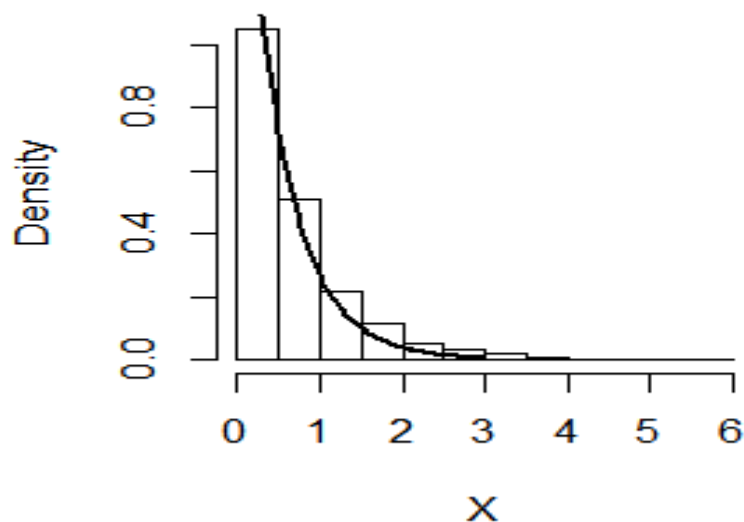
$$\therefore -\lambda x = \log(1 - y)$$

\therefore writing equation in terms of

$$\therefore \boxed{x = -\frac{\log(1 - y)}{\lambda}}$$

Putting values for y and λ as 1.5 for 1000 samples of inverse sampling method below histogram is plotted.

Inverse sampling of 1000 samples



Task C.3

The simple Bayesian network of Figure 1 has the joint probability distribution $p(\text{cloudy})p(\text{rain}|\text{cloudy})p(\text{sprinkler}|\text{cloudy})p(\text{wetgrass}|\text{sprinkler}, \text{rain})$. Use this to write a Gibbs Monte-Carlo sampler for the distribution. Run the sampler for 1000 cycles, throwing away the first 100 samples, and record counts for the two tables of $p(\text{wetgrass}, \text{cloudy})$ $p(\text{sprinkler}, \text{rain})$. Convert the counts to probabilities and report them.

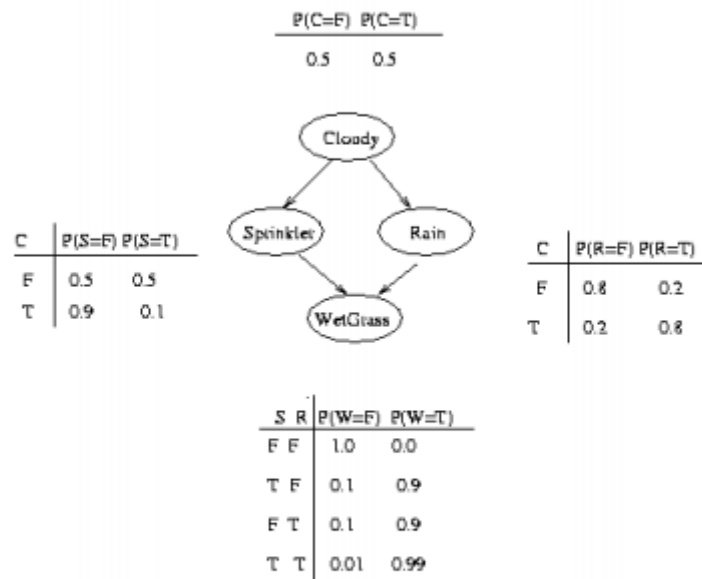


Figure 1: Simple Bayesian Network

For the given Bayesian Network, we can see that following points.

- The probability of cloudy is fair i.e. 0.5 for both true and false case.
- For Sprinklers, when cloudy is false, the probability of sprinklers is even 0.5. However, when Cloudy is true probability of sprinklers will be off is 0.9 while 0.1 probability for sprinklers being on.
- For Rain, when cloudy is false, the probability that it will rain is 0.2 and 0.8 probability that it won't rain. But when cloudy is true, the probability that it won't rain is 0.2 and 0.8 that it will rain.
- For wetgrass, the condition that sprinkler is off, and it isn't raining, then wetgrass is false with probability as 1. When sprinkler is true, and rain is off, the probability that grass will be wet is 0.9 and 0.1 of not being wet. When Sprinklers is on and its raining the probability that the grass is wet is 0.99 and 0.01 that it won't be wet.

From this above condition and considering only true values, the probability of $p(\text{wetgrass}|\text{cloudy})$ is 0.37 while the probability of $p(\text{sprinkler}|\text{rain})$ is 0.09.