CA-2

Bharat Jethwani

Student-ID: 10519364

Statistics for Data Analytics

Question 1:

DATASET:

Dataset used in Question 1 is Scrapped using python from Irish Times and ISE. Attaching the python scripts for the same in the Zip. I have merged all the scrapped data to create this dataset. Dataset contains information of ISEQ Overall performance from 2014 and all the PESTEL news published during that time frame. I have just taken the count of articles for GLM and applied linear GLM on it. It has various columns of ISEQ information, number of articles published and ISEQ\_Price is being used as Dependent variable.

Loading the data:

data1 <- read.csv("C:/Users/bhara/OneDrive/Desktop/ISEQOverallHistorical.csv" ,header = TRUE)  
data2 <- read.csv("C:/Users/bhara/OneDrive/Desktop/test.csv" ,header = TRUE)

Merging data in one data frame:

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(readr)  
data=full\_join(data1,data2)

## Joining, by = "Date"

## Warning: Column `Date` joining factors with different levels, coercing to  
## character vector

head(data)

## ï..ISEQ\_ID Date Price\_ISEQ Open\_ISEQ High\_ISEQ Low\_ISEQ Vol\_Mil\_ISEQ  
## 1 1 30-08-2019 5879.42 5846.54 5910.03 5846.54 26.37  
## 2 2 29-08-2019 5847.10 5754.21 5857.72 5748.84 24.30  
## 3 3 28-08-2019 5757.58 5831.34 5831.34 5735.47 30.60  
## 4 4 27-08-2019 5831.34 5744.73 5831.34 5709.75 32.58  
## 5 5 26-08-2019 5743.34 5750.06 5786.97 5716.39 8.93  
## 6 6 23-08-2019 5749.46 5770.87 5851.11 5739.23 36.17  
## ChangePercent\_ISEQ ï..Article\_Published  
## 1 0.55 4  
## 2 1.55 NA  
## 3 -1.26 3  
## 4 1.53 2  
## 5 -0.11 1  
## 6 -0.33 6

Replacing NA with mean values ignorning the outlier

summary(data$ï..Article\_Published)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 1.000 2.000 3.000 5.958 6.000 257.000 290

data3=data$ï..Article\_Published  
data1=na.omit(data$ï..Article\_Published)  
#summary(data1)  
bench=6+1.5\*IQR(data1) # getting outlier using third quartile  
data2=data1[data1<bench]  
#summary(data2)  
mea=mean(data2)  
data3[is.na(data3)]=mea  
#summary(data3)  
bench=5+1.5\*IQR(data3)  
data3[data1<bench]=mea  
summary(data3)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.000 3.675 3.675 3.723 3.675 31.000

Defining independent and dependent variables

y=data$Price\_ISEQ  
x1=data$Vol\_Mil\_ISEQ  
x2=data$ChangePercent\_ISEQ  
x3=data3  
x4=data$High\_ISEQ  
x5=data$Low\_ISEQ  
x6=data$Open\_ISEQ

Creating data frame

df=na.omit(data.frame(x1,x2,x3,x4,x5,x6,y))  
head(df)

## x1 x2 x3 x4 x5 x6 y  
## 1 26.37 0.55 3.675397 5910.03 5846.54 5846.54 5879.42  
## 2 24.30 1.55 3.675397 5857.72 5748.84 5754.21 5847.10  
## 3 30.60 -1.26 3.675397 5831.34 5735.47 5831.34 5757.58  
## 4 32.58 1.53 3.675397 5831.34 5709.75 5744.73 5831.34  
## 5 8.93 -0.11 3.675397 5786.97 5716.39 5750.06 5743.34  
## 6 36.17 -0.33 3.675397 5851.11 5739.23 5770.87 5749.46

Fitting the model

fit = glm(y ~., data=df, family='gaussian')  
summary(fit)

##   
## Call:  
## glm(formula = y ~ ., family = "gaussian", data = df)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -103.699 -3.351 0.194 3.532 42.803   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -5.128403 2.042952 -2.510 0.0122 \*   
## x1 0.008780 0.003617 2.428 0.0153 \*   
## x2 54.484421 0.454792 119.801 < 2e-16 \*\*\*  
## x3 0.065919 0.184838 0.357 0.7214   
## x4 0.069995 0.009041 7.742 1.84e-14 \*\*\*  
## x5 0.083990 0.006977 12.038 < 2e-16 \*\*\*  
## x6 0.846755 0.012952 65.374 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for gaussian family taken to be 60.31123)  
##   
## Null deviance: 720074097 on 1437 degrees of freedom  
## Residual deviance: 86305 on 1431 degrees of freedom  
## AIC: 9985  
##   
## Number of Fisher Scoring iterations: 2

Split the data into 80% as a trainset and 20% as a testset

set.seed(1225)  
n=nrow(df)  
indexes = sample(n,n\*(80/100))  
trainset = df[indexes,]  
testset = df[-indexes,]

Fit the full model

actual=testset$y  
full.model <- glm(trainset$y ~., data = trainset, family='gaussian')  
summary(full.model)

##   
## Call:  
## glm(formula = trainset$y ~ ., family = "gaussian", data = trainset)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -103.357 -3.339 0.216 3.585 42.784   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -4.724068 2.330146 -2.027 0.0429 \*   
## x1 0.008449 0.004178 2.022 0.0434 \*   
## x2 54.157330 0.505565 107.122 < 2e-16 \*\*\*  
## x3 0.207116 0.262589 0.789 0.4304   
## x4 0.074378 0.010168 7.315 4.83e-13 \*\*\*  
## x5 0.084172 0.007660 10.988 < 2e-16 \*\*\*  
## x6 0.842033 0.014444 58.295 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for gaussian family taken to be 61.66649)  
##   
## Null deviance: 592777529 on 1149 degrees of freedom  
## Residual deviance: 70485 on 1143 degrees of freedom  
## AIC: 8012.5  
##   
## Number of Fisher Scoring iterations: 2

Checking RMSE for Full Model

yhat=predict(full.model, testset[1:6])  
rmse\_f=sqrt((sum(yhat-actual)^2)/(nrow(testset)))  
rmse\_f

## [1] 9.510891

Reduced Model

library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

reduced.model=stepAIC(full.model)

## Start: AIC=8012.54  
## trainset$y ~ x1 + x2 + x3 + x4 + x5 + x6  
##   
## Df Deviance AIC  
## - x3 1 70523 8011.2  
## <none> 70485 8012.5  
## - x1 1 70737 8014.6  
## - x4 1 73785 8063.2  
## - x5 1 77931 8126.0  
## - x6 1 280043 9597.0  
## - x2 1 778120 10772.2  
##   
## Step: AIC=8011.16  
## trainset$y ~ x1 + x2 + x4 + x5 + x6  
##   
## Df Deviance AIC  
## <none> 70523 8011.2  
## - x1 1 70779 8013.3  
## - x4 1 73811 8061.6  
## - x5 1 77960 8124.5  
## - x6 1 280257 9595.9  
## - x2 1 778694 10771.1

yhat\_r=predict(reduced.model, testset[1:6])  
rmse\_r=sqrt((sum(yhat\_r-actual)^2)/(nrow(testset)))  
rmse\_r

## [1] 9.500048

Predictive Model

phat\_i=predict(full.model , testset, type='response')  
phat\_i

## 1 6 7 9 17 21 22 29   
## 5880.715 5756.008 5768.061 5692.163 5968.850 5980.425 6075.824 6331.378   
## 41 42 43 50 66 69 71 72   
## 6343.927 6422.679 6341.050 6069.662 6071.575 6150.201 6129.417 6150.527   
## 78 80 82 84 86 91 92 94   
## 6240.727 6161.487 6252.602 6400.482 6495.315 6397.152 6448.380 6533.306   
## 98 101 105 111 115 117 134 135   
## 6331.632 6253.464 6332.071 6029.865 6205.100 6252.026 6040.863 6056.318   
## 138 139 142 143 149 150 151 152   
## 5969.703 5953.076 5905.283 5834.016 5766.858 5857.335 5848.863 5841.311   
## 154 155 159 162 164 168 172 178   
## 5775.774 5832.348 5702.848 5668.112 5739.092 5667.960 5470.992 5420.436   
## 192 193 195 196 200 201 210 213   
## 5837.155 5869.854 5927.768 5855.019 5985.199 5967.206 6224.417 6097.379   
## 216 219 225 237 240 247 248 251   
## 5965.478 6026.288 6097.586 6572.884 6676.097 6614.830 6642.390 6682.251   
## 255 266 274 285 301 303 322 331   
## 6777.083 6665.769 6812.073 6938.722 6962.454 6979.465 7135.025 7007.717   
## 336 337 342 344 349 353 354 358   
## 6977.003 6937.718 6805.328 6801.270 6766.657 6639.976 6616.689 6637.556   
## 361 362 363 366 374 377 381 395   
## 6577.698 6590.010 6544.466 6507.048 6683.378 6722.371 6669.795 6605.304   
## 397 401 402 403 407 415 427 435   
## 6742.165 6949.912 6968.198 6994.085 7170.748 7067.980 7016.749 7020.467   
## 438 441 447 450 455 458 462 478   
## 6993.640 6944.891 6871.107 6947.689 6763.225 6928.222 6979.503 6819.760   
## 485 489 495 496 509 511 512 515   
## 6907.852 6783.552 6683.460 6680.130 6547.837 6607.337 6651.771 6727.951   
## 516 518 522 524 526 530 532 535   
## 6677.701 6793.613 6581.823 6626.964 6687.828 6687.492 6673.457 6695.966   
## 538 540 542 546 548 552 559 570   
## 6788.642 6870.477 6893.872 6915.028 6860.573 6829.774 7089.146 6982.788   
## 574 580 583 593 600 608 609 610   
## 6988.626 6933.605 6912.618 6986.206 6739.756 6706.825 6666.348 6648.253   
## 612 613 617 619 623 631 634 635   
## 6618.129 6651.900 6565.099 6638.665 6712.142 6602.114 6678.120 6692.541   
## 641 646 649 654 655 662 666 697   
## 6620.650 6531.414 6471.943 6470.632 6453.766 6597.876 6569.237 6221.865   
## 701 709 718 728 733 738 740 744   
## 6291.182 6276.291 5865.555 6005.544 5864.391 6068.059 6025.828 6018.745   
## 745 748 755 760 761 766 773 788   
## 6109.911 6078.498 6191.171 6293.880 6190.695 6120.886 6030.278 5788.636   
## 793 801 812 817 823 826 827 828   
## 5805.770 5503.850 6287.552 6008.072 6469.942 6437.642 6495.641 6495.246   
## 829 834 838 840 844 845 846 847   
## 6496.773 6226.593 6154.256 6143.749 6038.186 6039.518 6020.408 6047.476   
## 858 868 870 885 887 896 898 899   
## 6213.004 6281.703 6337.013 6311.945 6349.995 6124.201 6109.477 5925.512   
## 906 909 912 916 919 920 923 926   
## 6082.613 6353.046 6217.156 6353.242 6362.647 6287.367 6621.309 6572.476   
## 927 928 932 933 936 956 959 965   
## 6570.194 6690.564 6852.710 6830.341 6699.150 6652.496 6676.860 6560.116   
## 973 980 984 990 996 997 1003 1006   
## 6424.678 6352.503 6228.993 6187.144 6063.824 6231.955 6469.814 6466.187   
## 1015 1016 1018 1021 1030 1034 1035 1039   
## 6272.589 6266.167 6308.432 6097.096 6390.166 6617.912 6697.042 6450.055   
## 1040 1043 1047 1048 1052 1059 1062 1083   
## 6425.851 6485.586 6498.213 6483.119 6376.068 6247.793 6214.225 6360.210   
## 1087 1097 1106 1110 1121 1125 1133 1142   
## 6270.317 6225.332 6297.062 6257.412 6018.982 5997.327 6119.343 5895.173   
## 1155 1165 1169 1173 1183 1188 1189 1198   
## 5687.375 5494.283 5591.366 5332.874 5158.655 5196.798 5233.805 5147.810   
## 1206 1217 1219 1229 1241 1244 1249 1250   
## 5070.849 4776.523 4831.701 4695.241 4468.872 4665.363 4887.058 4816.133   
## 1257 1266 1284 1285 1291 1293 1299 1305   
## 4929.684 4953.056 4599.355 4603.437 4615.823 4711.053 4734.800 4714.035   
## 1314 1319 1323 1332 1334 1349 1351 1356   
## 4761.874 4738.649 4788.671 4954.417 4897.744 4899.594 4939.514 4898.859   
## 1365 1373 1378 1381 1382 1388 1395 1407   
## 4840.276 5061.085 4965.884 4836.550 4921.899 4960.762 5097.995 4933.239   
## 1410 1414 1431 1432 1433 1434 1435 1438   
## 4860.527 4699.782 4803.792 4733.330 4757.375 4708.957 4704.086 4557.756

Since Linear GLM Model is used there won’t be any Confusion Matrix. For Accuracy Using RMSE

Accuracy

RMSE=c(0,0)  
RMSE=RMSE+c(rmse\_f,rmse\_r)  
RMSE

## [1] 9.510891 9.500048

**Question 2**

Let are identically independently distributed (iid) with Poisson().

Compute the likelihood function (LF)

A close up of text on a white background

Description automatically generated

A close up of a piece of paper

Description automatically generated b) Adopt the appropriate conjugate prior to the parameter (Hint: Choose hyperparameters optionally within the support of distribution).

C) Using (a) and (b), find the posterior distribution of

A close up of text on a black background

Description automatically generated

d) Compute the minimum Bayesian risk estimator of . A close up of a piece of paper

Description automatically generated **Question 3**

A close up of text on a white background

Description automatically generated

