# ACSC/STAT 3740, Predictive Analytics

WINTER 2024
Toby Kenney

Homework Sheet 3

Model Solutions

## Standard Questions

1. *An online shopping company is building a recommendation system to suggest purchases. It has collected the following data in the file* ***HW3Q1.txt***.

| *Variable* | *Meaning* |
|---|---|
| *category* | *The category of the item* |
| *market* | *The buyers targetted by the product.* |
| *brand.quality* | *The brand quality of the item: 0=no branding, 1=top quality* |
| *popularity* | *A measure of the number of users who purchase the item.* |
| *user.ID* | *A unique identifier for each user in the system.* |
| *month* | *The month of the year.* |
| *recent.spending* | *The amount spent by the user in the past month.* |
| *price* | *The price of the product.* |
| *purchase* | *Whether the user purchases the item.* |

*(a) Fit a random forest to predict whether the user will purchase the item. Use this to predict the probability that the user will purchase the item for the cases in the file* ***HW3Q1_test.txt***.

```
 library(caret)

 library(dplyr)

RF.model<-train(HW3Q1%>%select(-c("purchase")),HW3Q1$purchase,method="rf",
                   trControl=trainControl(method="repeatedcv",
                                          number=10,
                                          repeats=2),
                   tuneGrid=expand.grid(mtry=seq_len(8)),
                   ntree=500)


RF.predict<-predict(RF.model,newdata=HW3Q1_test,type="prob")
```

This predicts the following probabilities:

| Case | Purchase Probability | Case | Purchase Probability |
|---|---|---|---|
| 1 | 0.564 | 8 | 0.192 |
| 2 | 0.764 | 9 | 0.460 |
| 3 | 0.596 | 10 | 0.618 |
| 4 | 0.636 | 11 | 0.650 |
| 5 | 0.528 | 12 | 0.388 |
| 6 | 0.004 | 13 | 0.734 |
| 7 | 0.654 | 14 | 0.136 |

*(b) A natural feature to add to this data set is the ratio* `price/recent.spending`*. Refit the random forest predictor with this feature added, and use this to estimate the probabilities for the test data.*

```
RF.model2<-train(HW3Q1%>%mutate(relative.price=price/recent.spending)%>%select(-c("purcha
                 trControl=trainControl(method="repeatedcv",
                                        number=10,
                                        repeats=2),
                 tuneGrid=expand.grid(mtry=seq_len(9)),
                 ntree=500)


RF.predict2<-predict(RF.model2,HW3Q1_test%>%mutate(relative.price=price/recent.spending),
```

This predicts the following probabilities:

| Case | Purchase Probability | Case | Purchase Probability |
|---|---|---|---|
| 1 | 0.544 | 8 | 0.252 |
| 2 | 0.694 | 9 | 0.498 |
| 3 | 0.686 | 10 | 0.590 |
| 4 | 0.612 | 11 | 0.658 |
| 5 | 0.540 | 12 | 0.382 |
| 6 | 0.022 | 13 | 0.730 |
| 7 | 0.676 | 14 | 0.152 |

2. *The file* `HW3Q2.txt` *contains data from a study on the corrosion of alloys under various conditions. The variables included are*

| *Variable* | *Meaning* |
|---|---|
| *iron.percent* | *The percentage of iron in the sample.* |
| *chrome.percent* | *The percentage of chrome in the sample.* |
| *temperature* | *The temperature at which the sample is kept.* |
| *humidity* | *The humidity at which the sample is kept.* |
| *salt.concentration* | *The concentration of salt in the air around the sample.* |
| *light.intensity* | *The intensity of light to which the sample is exposed.* |
| *sample.impurity* | *The percentage of impurities in the sample.* |
| *one.hour.oxidation* | *The percentage of the sample which oxidises in a 1-hour period.* |
| *ten.hour.oxidation* | *The percentage of the sample which oxidises in a 10-hour period.* |

*Fit a generalised linear model to predict whether the one hour oxidation and ten hour oxidation will be non-zero, and conditional on these being non-zero, fit a GLM with a gamma distribution for the conditional distribution of the one-hour and ten-hour oxidisation. Use this to predict the oxidation for the data in `HW3Q2_test.txt`.*

```
HW3Q2<-read.table("HW3Q2.txt")
 HW3Q2_test<-read.table("HW3Q2_test.txt")

GLM.model.one<-glm(one.hour.oxidation >0~.-ten.hour.oxidation ,data=HW3Q2,family=binomial(l



GLM.model.ten<-glm((ten.hour.oxidation >0)~.-one.hour.oxidation ,data=HW3Q2,family=binomial


 library(dplyr)

GLM.model.one.condit<-glm(one.hour.oxidation~.-ten.hour.oxidation ,data=HW3Q2%>%filter(one



GLM.model.ten.condit<-glm(ten.hour.oxidation~.-one.hour.oxidation ,data=HW3Q2%>%filter(one

GLM.one.predict<-predict(GLM.model.one,
                         newdata=data.frame(HW3Q2_test,
                                            "ten.hour.oxidation"=as.numeric(NA)),
                         type="response")
### My version of R has a bug that requires the existence of an unused
### variable of type numeric to make predictions.


GLM.one.condit.predict<-predict(GLM.model.one.condit,
                                newdata=data.frame(HW3Q2_test,
                                                   "ten.hour.oxidation"=as.numeric(NA)),
                         type="response")
GLM.ten.predict<-predict(GLM.model.ten,
                         newdata=data.frame(HW3Q2_test,
                                            "one.hour.oxidation"=as.numeric(NA)),
                         type="response")
GLM.ten.condit.predict<-predict(GLM.model.ten.condit,
                                newdata=data.frame(HW3Q2_test,
                                                   "one.hour.oxidation"=as.numeric(NA)),
                         type="response")
                                        #
##display results in a table
 cbind(GLM.one.predict        ,GLM.one.condit.predict,GLM.ten.predict
 ,GLM.ten.condit.predict)
```

This predicts the following

| Sample | One Hour Oxidation | | Ten Hour Oxidation | |
|---|---|---|---|---|
| | Probability | Conditional Expectation | Probability | Conditional Expectation |
| 1 | 0.12991784 | 0.08964353 | 1 | 0.2481112 |
| 2 | 0.74330800 | 0.11878600 | 1 | 0.2617243 |
| 3 | 0.63658770 | 0.12509734 | 1 | 0.2724315 |
| 4 | 0.21931215 | 0.10174189 | 1 | 0.2564783 |
| 5 | 0.12175742 | 0.08869116 | 1 | 0.2325418 |
| 6 | 0.06194273 | 0.08284463 | 1 | 0.2146201 |
| 7 | 0.07912872 | 0.08937795 | 1 | 0.2333065 |
| 8 | 0.22009041 | 0.09694738 | 1 | 0.2536921 |
| 9 | 0.56980307 | 0.12520152 | 1 | 0.2953257 |
| 10 | 0.98674738 | 0.18974964 | 1 | 0.4598823 |
| 11 | 0.17999166 | 0.09070002 | 1 | 0.2340893 |

3. *The file* **`HW3Q3.txt`** *contains daily maximum temperature recordings in a certain city.*

   *(a) Fit a seasonal trend using the function* $\sin(2\pi t)$ *and* $\cos(2\pi t)$ *where* $t$ *is the time in years, and a linear trend to reflect global warming.*

```
HW3Q3<-read.table("HW3Q3.txt")

### Need to convert to date.

HW3Q3.series<-data.frame(
     date=as.Date(paste(HW3Q3$year,HW3Q3$month,HW3Q3$day,sep="-"),
                    format="%Y-%b-%d"), #4-digit year-month abbreviation-day
     temperature=HW3Q3$temperature)


library(dplyr)


trend<-lm(temperature~sin(2*pi*t)+cos(2*pi*t)+t,
                    data=HW3Q3.series%>%
                         mutate(
                              t=as.numeric(date)/365.25)) # convert time to years.

summary(trend)
```

gives the following trend:

| | Estimate | Std. Error | $t$ value | Pr(> $\|t\|$) |
|---|---|---|---|---|
| (Intercept) | −21.42710 | 1.16460 | −18.40 | $< 2 \times 10^{-16}$ |
| $\sin(2\pi t)$ | −3.85592 | 0.14893 | −25.89 | $< 2 \times 10^{-16}$ |
| $cos(2\pi t)$ | −12.66148 | 0.14773 | −85.70 | $< 2 \times 10^{-16}$ |
| $t$ | 0.63380 | 0.02479 | 25.57 | $< 2 \times 10^{-16}$ |

*(b) After subtracting the seasonal and linear trends, fit an ARMA model to the residuals, using AIC to determine the best choices for p and q.*

```
library(forecast)
temp.resid.arma<-auto.arima(trend$residuals,ic="aic",max.d=0)

summary(temp.resid.arma)
```

This selects an ARMA(4,3) model with the following coefficients:

| Coefficient | Estimate | Standard Error |
|---|---|---|
| ar1 | −0.8028 | 0.0616 |
| ar2 | 0.2632 | 0.0560 |
| ar3 | 0.1199 | 0.0404 |
| ar4 | −0.0106 | 0.0373 |
| ma1 | 1.1004 | 0.0601 |
| ma2 | 0.6473 | 0.0597 |
| ma3 | 0.3469 | 0.0270 |

*(c) Fit a GARCH model to model the variance.*

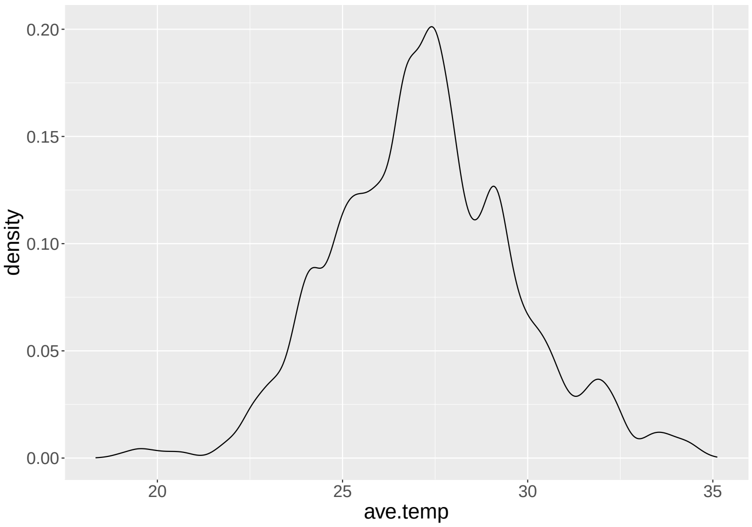
```
library(rugarch)
GARCH_model<-ugarchspec(mean.model=list(armaOrder=c(4,3)), distribution="norm")
GARCH_temp<-ugarchfit(GARCH_model,trend$residuals,solver="hybrid")
## The default solver fails to converge.
GARCH_temp
```

This fits the following parameters:

| Parameter | Estimate | Standard Error | $t$-value | $p$-value |
|---|---|---|---|---|
| mu | 0.000000 | 0.177248 | 0.00000 | 1.000000 |
| ar1 | −0.805374 | 0.061675 | −13.05829 | 0.000000 |
| ar2 | 0.265061 | 0.056212 | 4.71538 | 0.000002 |
| ar3 | 0.121897 | 0.040497 | 3.01004 | 0.002612 |
| ar4 | −0.011859 | 0.037311 | −0.31784 | 0.750604 |
| ma1 | 1.102171 | 0.060121 | 18.33244 | 0.000000 |
| ma2 | 0.645831 | 0.060027 | 10.75895 | 0.000000 |
| ma3 | 0.345902 | 0.027060 | 12.78261 | 0.000000 |
| omega | 8.751903 | 9.637000 | 0.90816 | 0.363796 |
| alpha1 | 0.006389 | 0.009543 | 0.66951 | 0.503171 |
| beta1 | 0.746949 | 0.274558 | 2.72054 | 0.006517 |

*(d) Based on this model, what is the probability that the average temperature in July 2026 will exceed 30°C? [You can use the `ugarchboot` function to run a simulation to estimate this.]*

```
GARCH_Bootstraps<-ugarchboot(GARCH_temp,
                             method=”full”,
                             n.ahead=as.Date(”2026-07-31”)-as.Date(”2024-02-12”),
                             n.bootfit=400, # 1000 parameter estimates
                             n.bootpred=400, # 1000 bootstraps
                             rseed=seq_len(800)) #Need to explicitly set seed
### rseed needs to be a vector of length n.bootfit+n.bootpred

### This may take a few minutes to run. To make it run faster, you
### could reduce n.bootfit to about 100.  You could also use
### 'method=”partial”' to used fixed parameter estimates from
### part (b).


### Calculate Distribution of average temperature
GARCH_boot_july<-GARCH_Bootstraps@fseries[,(as.Date(”2026-07-01”)-as.Date(”2024-
02-12”)):(as.Date(”2026-07-31”)-as.Date(”2024-02-12”))]
july_trend<-predict(trend,newdata=list(”t”=(as.numeric(as.Date(”2026-07-01”)):as.numeric(
ave.temp<-rowMeans(GARCH_boot_july+
                   rep(1,dim(GARCH_boot_july)[1])%*%t(july_trend))
### Remember to add the trend.
### It is possible that parameter estimates do not converge for some simulations
### So use dim(GARCH_boot_july)[1] instead of 160000

ggplot(data.frame(”ave.temp”=ave.temp),mapping=aes(x=ave.temp))+geom_density()+largertext


mean(ave.temp>30)
### probability of average temperature exceeding 30.
```



In my bootstap, the probability of this event is 0.123025.

4. *A provincial government has collected the following data on the effect of regulations on economic activity in the file* `HW3Q4.txt`*.*

| *Variable* | *Meaning* |
|---|---|
| *interest.rates* | *The prime interest rates* |
| *unemployment* | *The unemployment rate* |
| *consumer.confidence* | *An index measuring consumer confidence* |
| *CPI* | *Annual consumer price inflation over the previous 12 months* |
| *stock.index.returns* | *The increase in the stock market index over the past 12 months* |
| *safety.regulations* | *An index measuring the number of safety regulations in place for businesses* |
| *other.regulations* | *An index measuring the number of non-safety regulations in place for businesse* |
| *GDP* | *The GDP growth during the following 12 months* |

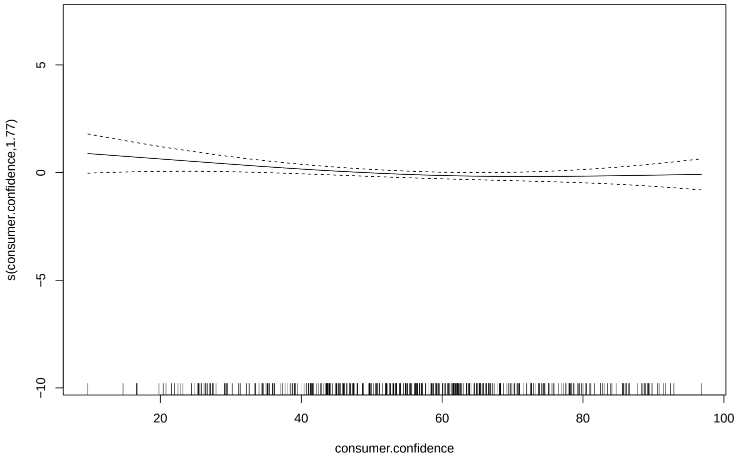
*Fit a generalised additive model to predict the GDP growth, using a normal response variable and identity link function.*

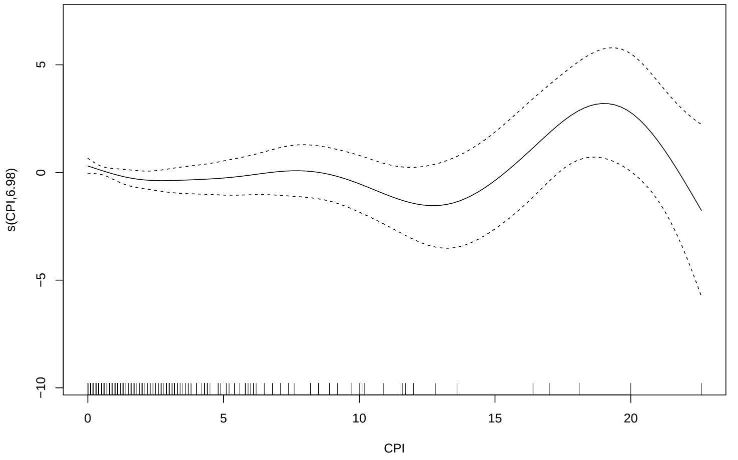
*Use this model to predict GDP growth for the cases in the file* `HW3Q4_test.txt`*.*

```
HW3Q4<-read.table("HW3Q4.txt")
HW3Q4_test<-read.table("HW3Q4_test.txt")

library(mgcv)

GAM_model<-gam(GDP~., data=HW3Q4)
summary(GAM_model)
```

This fits a linear model in the variables `interest.rates`, `unemployment`, `stock.index.returns` and `other.regulations`, a nearly-linear model in `safety.regulations` and non-linear models in `consumer.confidence` and `CPI`.

The fitted non-linear functions of `consumer.confidence` and `CPI` are shown in the following plots:





The predictions for the test data are as follows:

| Test data | Predicted GDP | Test data | Predicted GDP | Test data | Predicted GDP |
|---|---|---|---|---|---|
| 1 | 1.2139332 | 5 | 1.6807840 | 9 | 0.4446145 |
| 2 | 1.8921183 | 6 | 0.2810598 | 10 | 0.1482492 |
| 3 | −0.1090172 | 7 | 7.1063137 | 11 | 0.4910823 |
| 4 | 0.9182445 | 8 | 0.7151467 | 12 | 1.4832226 |

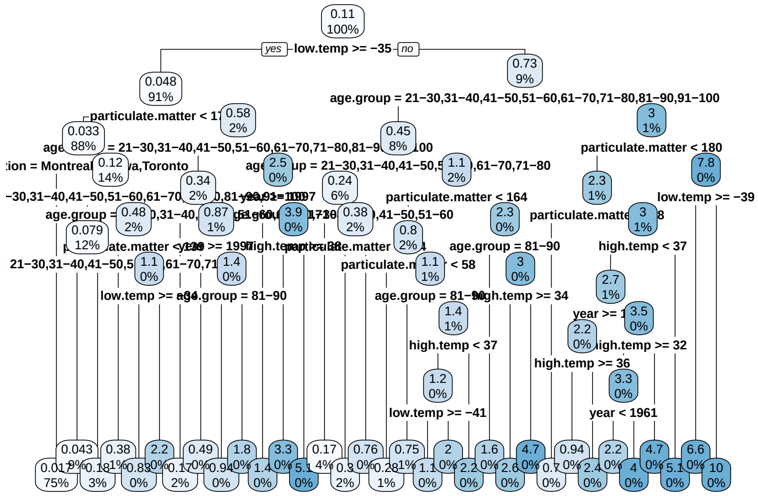
5. *A life insurance company has collected the following data on the effect of particulate matter pollution on mortality. The data are in the file* ***HW3Q5.txt****.*

| *Variable* | *Meaning* |
|---|---|
| *year* | *The year of the data* |
| *age.group* | *The age range of the population in question* |
| *location* | *The city being studied.* |
| *high.temp* | *The highest temperature during the year* |
| *low.temp* | *The lowest temperature during the year* |
| *particulate.matter* | *The average amount of particulate matter in the air* |
| *mortality* | *The percentage of this age group who died during the year.* |

*(a) Fit a decision tree to predict mortality from the other variables.*

```
HW3Q5<−read.table("HW3Q5.txt",stringsAsFactors=TRUE)
HW3Q5_test<−read.table("HW3Q5_test.txt",stringsAsFactors=TRUE)

library(rpart.plot)

HW3Q5_dt<−rpart(mortality~.,data=HW3Q5,control=rpart.control(minbucket=1,cp=1e−20,xval=1

HW3Q5_cp_1se_error<−min(HW3Q5_dt$cptable[,4]+HW3Q5_dt$cptable[,5])
HW3Q5_cp_1se<−HW3Q5_dt$cptable[min(which(HW3Q5_dt$cptable[,4]<HW3Q5_cp_1se_error)),1]
HW3Q5_dt_1se<−prune(HW3Q5_dt,cp=HW3Q5_cp_1se)
rpart.plot(HW3Q5_dt_1se)
```

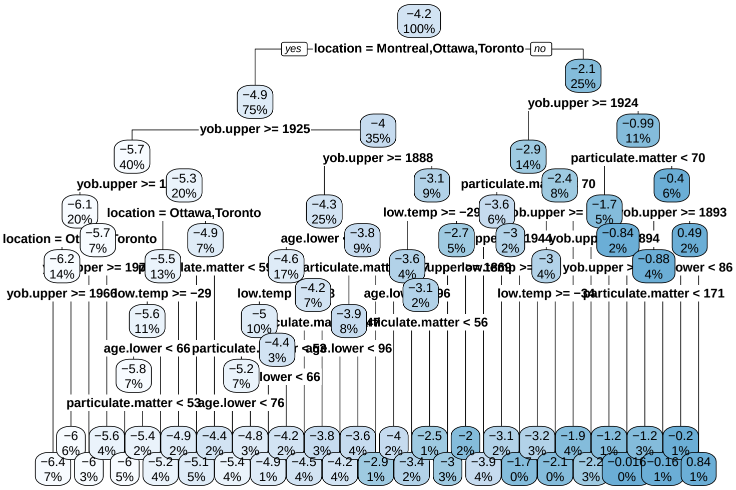
The cross-validated error keeps falling so we use the 1-s.e. cut-off to avoid the tree becoming too complicated.



*(b) Convert age group to numeric, add year of birth as a predictor, and log-transform the mortality rate, and refit a decision tree.*

```
### First split the age group into numerical lower and upper bounds.

HW3Q5_tidy<-HW3Q5%>%mutate(age.group.corrected=
                                    recode_factor(HW3Q5$age.group,
                                                    "over 100"="100-120"))%>%
    select(-c("age.group"))%>%
    separate(col="age.group.corrected",into=c("age.lower","age.upper"))

HW3Q5_tidy$age.lower<-as.integer(HW3Q5_tidy$age.lower)
HW3Q5_tidy$age.upper<-as.integer(HW3Q5_tidy$age.upper)

HW3Q5_yob<-HW3Q5_tidy%>%mutate(yob.upper=year-age.lower)

HW3Q5_dt_yob<-rpart(log(mortality)~.,data=HW3Q5_yob,control=rpart.control(minbucket=1,cp=

HW3Q5_yob_cp_1se_error<-min(HW3Q5_dt_yob$cptable[,4]+HW3Q5_dt_yob$cptable[,5])
HW3Q5_yob_cp_1se<-HW3Q5_dt_yob$cptable[min(which(HW3Q5_dt_yob$cptable[,4]<HW3Q5_yob_cp_1
HW3Q5_yob_dt_1se<-prune(HW3Q5_dt_yob,cp=HW3Q5_yob_cp_1se)

### Tree is too complicated, so show only the top
rpart.plot(prune(HW3Q5_dt_yob,cp=0.001))
```

The cross-validated error becomes much smaller on the log scale, but even at the 1-s.e. cut-off, the fitted tree is very complicated. We plot only the top part of the tree by pruning much further.



*(c) Fit a random forest model to predict log mortality from the other variables, including year of birth. Use this model to predict mortality for all age groups in Toronto in 2025 if the high temperature is 37°C, the low temperature is −11°C, and particulate matter is 133.*

```
library(caret)
library(dplyr)

RF.model<-train(HW3Q5_yob%>%select(-c("mortality")),
                log(HW3Q5_yob$mortality),
                method="rf",
                trControl=trainControl(method="repeatedcv",
                                       number=10,
                                       repeats=2),
                tuneGrid=expand.grid(mtry=seq_len(8)),
                ntree=500)

### This takes a little while to run. You may need to reduce ntree to
### get the results.

HW3Q5_test<-data.frame(
    "year"=rep(2025,9),"age.lower"=seq_len(9)*10+11,"age.upper"=seq_len(9)*10+20,"locatio
)

RF.predict<-exp(predict(RF.model,newdata=HW3Q5_test))
### remember that we log-transformed the response, so we need to exponentiate to get the

### Display results in a nice table.
cbind(paste(HW3Q5_test$age.lower,HW3Q5_test$age.upper,sep="-"),round(RF.predict,5))
```

In my fitting, this predicts the following mortalities for each age group:

| Age group | Predicted Mortality |
|---|---|
| 21-30 | 0.00300 |
| 31-40 | 0.00316 |
| 41-50 | 0.00370 |
| 51-60 | 0.00433 |
| 61-70 | 0.00553 |
| 71-80 | 0.00704 |
| 81-90 | 0.01279 |
| 91-100 | 0.02013 |
| over 100 | 0.02781 |

[As random forest has some randomness, results may vary slightly.]