Predicting the Present with Google Trends

Forcasting Flu Season

STA 237A Final Project

Kenneth Broadhead, Xuezhen Li

December 13, 2019

1 Abstract

In this paper, we consider using google trends search data to predict current, but presently unknown values, of time series data of interest. Two examples are discussed, an economic example involving automobile sales, and a public health example involving seasonal flu data collected from the CDC. Potential benefits and limitations of this method of "forecasting the present" are also discussed.

2 Introduction

Economic reports for a given time period are plagued by a significant reporting lag. Similarly, weekly flu Data from the CDC's network of Hospitals and Clinics takes time to compile and release to the public. Understandably, economic researchers and public health officials, respectively, would like to have access to these reports much sooner. However, the reporting delay in both cases is due to the fact that these data sets are large and complex, taking time compile and publish. Reporting delay is often simply unavoidable.

However, Google Trends data for search queries related to the information contained in these reports is available in almost real time. It is not unreasonable to suspect that these search queries are correlated with the subject matter of the reports of interest, for the search data is reflective of the actions, (or experiences) of those about whom the reports are in some sense about. We therefore endeavor to investigate whether it is possible to predict the current, but yet to be released, values of these reports using current Google Trends data. We first explain to the reader relevant aspects of Google Trends, we then explore two examples: predicting automobile sales, and predicting the national prevalence of flu in the US.. We close with a discussion of the potential benefits and limitations of this method of "predicting the present," as well as possible methods for improvement that should be investigated further.

3 Google Trends

Google Trends provides a time series index of the volume of queries users enter into Google in a given geographic area. Figure 1 shows the search index for [UC Davis] in the United States in the past five years (As a side note, we the sudden spike in interest in UC Davis occurred during March madness, when UC Davis competed in the NCAA men's basketball tournament). The maximum query share in the time period specified is normalized to be 100 and the query share at the initial date being examined is normalized to be zero. In addition to the region and period of time, we can also restrict the search index to a specific category. Google classifies search queries into about 30 categories at the top level and about 250 categories at the second level using a natural language classification engine.

The last thing we mention, is the limitation of acquiring Google Trends data. The longer the time period you choose, the longer the time interval between two consecutive search indexes. For example, if the time period is longer than five years, you can only get the monthly Google trends data instead of weekly data.

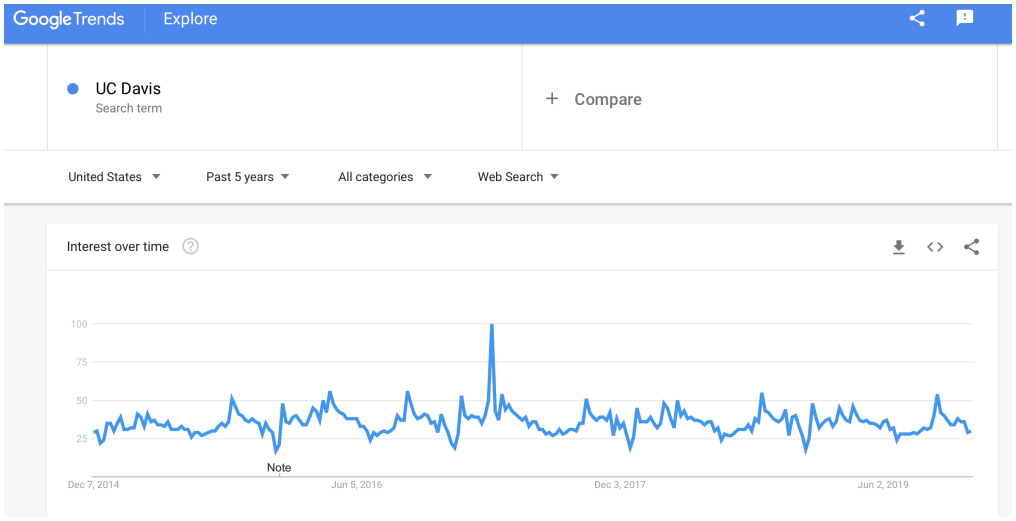

Figure 1: Search index for [UC Davis] in the United States

4 Economic Data Example

As the first economic data example we use the "Motor Vehicles and Parts Dealers" series from the U.S. Census Bureau "Advance Monthly Sales for Retail and Food Services" report.[1] Figure 2 shows the data from 2005 to 2011 after the log transformation.

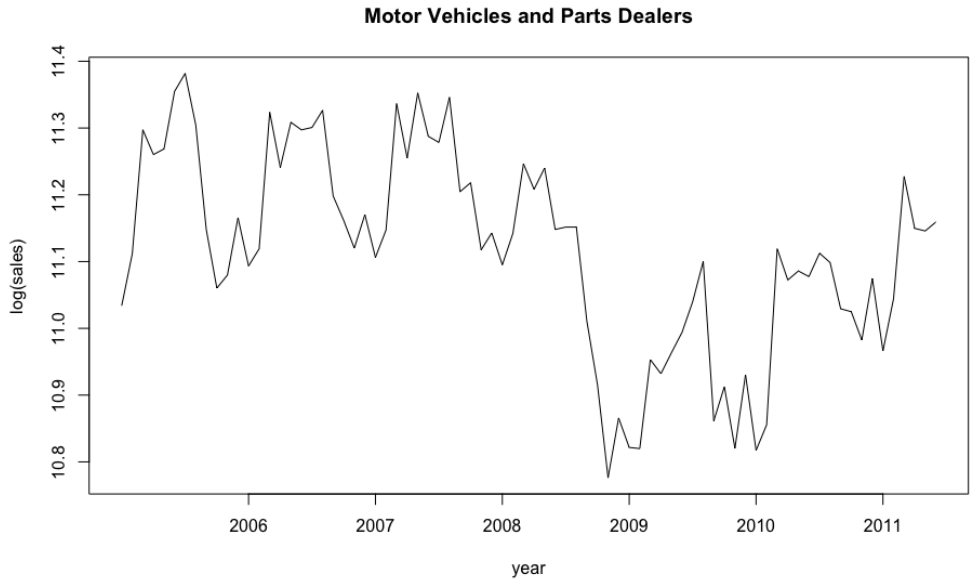

Figure 2: log(sales) of Motor Vehicles and Parts

By looking at the ACF plot we decide to choose AR(5) seasonal model as the initial full model and use all subset regression method to implement model selection. Figure 3 shows that the AR(1) seasonal model is the best model based on BIC and Mallows' C_p criteria. The R^2 of the AR(1) seasonal model is 0.7211.

lag1	lag2	lag3	lag4	lag5	lag12	R^2	adj R^2	Cp	BIC
*						0.6626	0.6581	15.9899	-76.0299
*					*	0.7211	0.7137	2.3798	-86.5332
*			*		*	0.7246	0.7135	3.4406	-83.1677
*	*		*		*	0.7328	0.7182	3.2638	-81.1579
*	*	*	*		*	0.7336	0.7151	5.0420	-77.0443
*	*	*	*	*	*	0.7338	0.7113	7.0000	-72.7337

Figure 3: all subset regression

The base model is

$$y_t = b_0 + b_1 y_{t-1} + b_{12} y_{t-12} + e_t \qquad \text{(base model)}$$

Google Trends contains several automotive-related categories. We add two automotive-related categories from the Google trends.[1] One is Trucks and SUVs, the other one is Auto Insurance. To check the temporal aspects of the relationship between response and google trends covariates, we plot two standardized series in the same graph. Figure 4 shows in most of the time period, google trends variables have high correlations with the sales data.

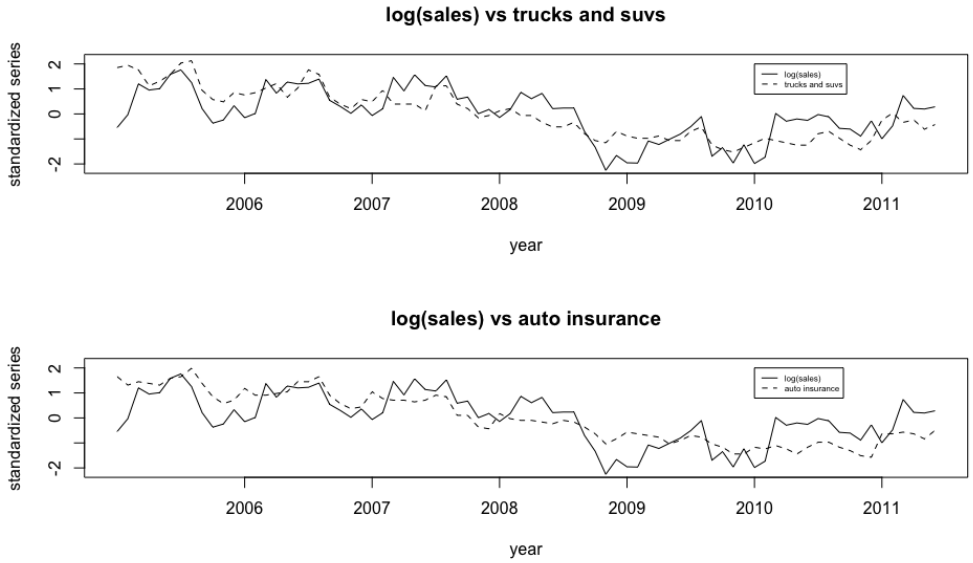

Figure 4: temporal aspects relationship

Therefore, the google trends model becomes

$$y_t = b_0 + b_1 y_{t-1} + b_{12} y_{t-12} + a_1 g_{1,t} + a_2 g_{2,t} + e_t \qquad \text{(google trends model)}$$

The R^2 of the google trends model is 0.7821, which suggests the google trends variables significantly improve in-sample fit when added to this regression. Moreover, the diagnosis of the google trends model is based on the graphs of residuals.

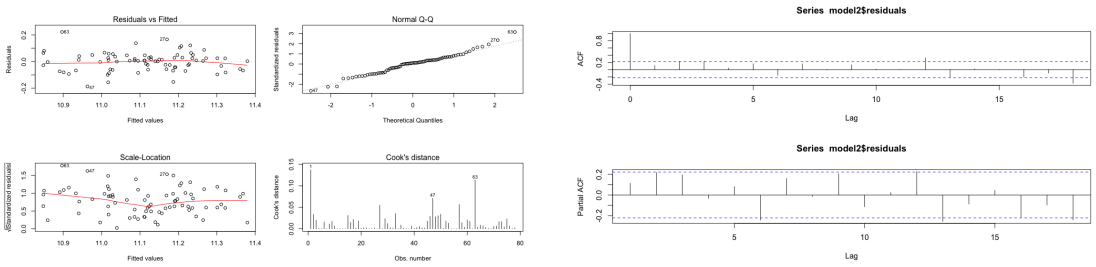

Figure 5: Diagnosis of the google trends model

It seems that the residual is the white-noise series with normal distribution, indicating the google trends model is a suitable model in this case. The further question of interest is whether the trends variables improve out-of-sample forecasting. To check this, we use a rolling window forecast. We set the length of each rolling window to be two-year and make an one-step ahead prediction. Then we compute the MAE for each model and make a comparison. The results are shown in figure 6. The overall MAE decreases from 6.38% to 5.71%. However, if we just examine the MAE especially in the recession period, it decreases from 7.61% to 5.19%.

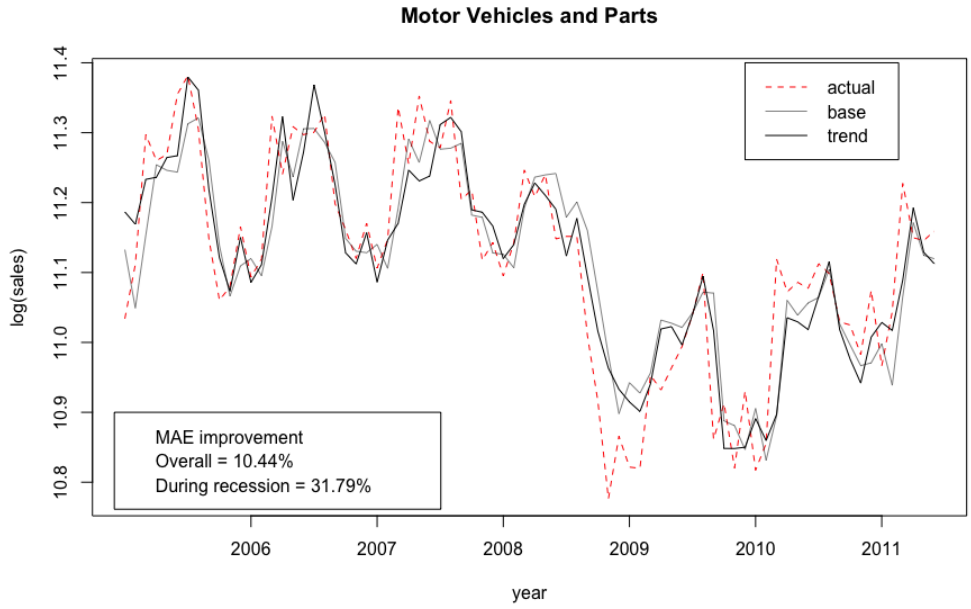

Figure 6: Motor Vehicles and Parts

So far we notice google trends can indeed help us predict the present. The last question of interest in this section is how well the google trends model predict the present. To investigate this question, we try to leave out the end of the actual data set and use Google trends to make one-step ahead prediction for the period 2005-06-01 to 2011-06-01. Figure 7 shows the prediction is quite well in most of the period. It may give some indication of the peak points and change points. The worst predicted period is the end of the 2009 year. Figure 4 may explain part of the reason. In that period the relationship between google trends and sales is not really strong probably due to some unknown events. But overall the google trends predict the present sales quite well.

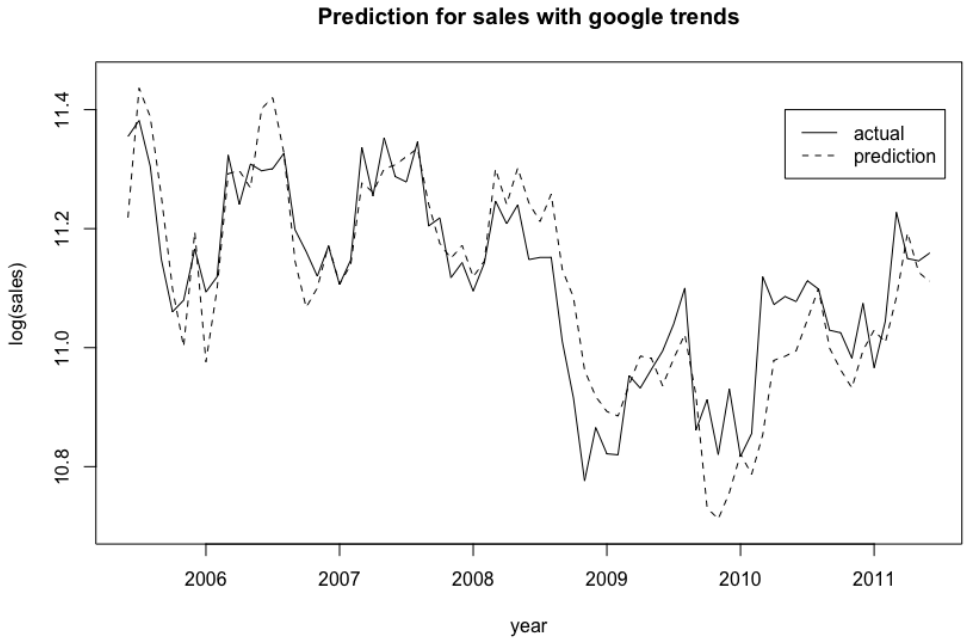

Figure 7: Prediction for sales with google trends

5 CDC Flu Data Example

Each week, the CDC estimates the national percentage of visits to healthcare providers for influenza like illnesses. This measure (denoted ILINet) is used to assess questions of flu season start, end, intensity, peak times, etc.... Understandably, having the most up to data information on ILINet is a priority; but, given the complexity of measuring ILINet, reporting delays are inevitable. However, perhaps using Google Trends data for the previous week, the most up-to-date value of ILINet can be predicted. We provide a graph of the relevant data in the figure 8.

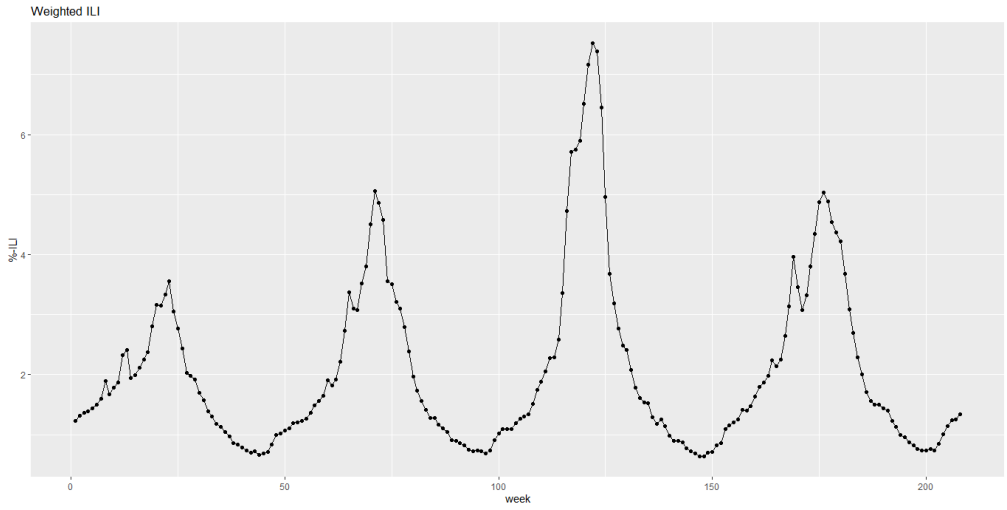

Figure 8: FLU Data

Inspection of the graph suggests that a seasonal AR model might be appropriate. The ACF and PACF plots (shown in the figure 9) further indicate that a seasonal AR(2) model should be adequate. Furthermore, running a Box-Cox procedure (not shown) suggests heavily that an inverse transformation is needed. We thus fit the seasonal AR(2) model $x_t = b_0 + b_1 x_{t-1} + b_2 x_{t-2} + b_{52} x_{t-52} + e_t$ to the inverse of the ILINet data. The model provides a very good fit, with an adjusted R^2 of 0.9833.

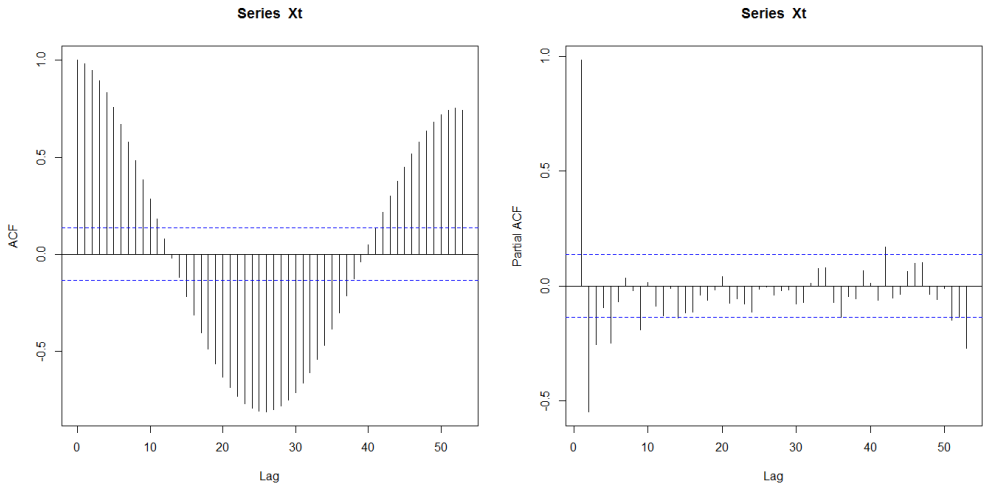

Figure 9: ACF and PACF plot

There are many searches that could be indicative of flu activity, such as "flu season," "flu shots," "flu symptoms," and many more. We considered a best subsets routine for models including "flu season," "flu shots," "flu symptoms," and "flu" search query data (for the U.S.) as exogenous predictors, as well as the lag values assumed in the base model. The routine suggested that the model utilizing all but the "flu shots" data gives the best possible model he best model by AIC and adjusted R^2 criterion. See the code supplied in the appendix . We therefore fit a seasonal AR(2) model with these three Google Search terms as exogenous predictors. The model equation is $x_t = b_0 + b_1 x_{t-1} + b_2 x_{t-2} + b_5 2 x_{t-52} + a_1 g_{1,t} + a_2 g_{2,t} + a_3 g_{3,t} e_t$ (The BIC criterion suggests that only the "flu shots" data should be used). The Adjusted R^2 for this new model is 0.9842.

We note that there is only a very small improvement to in-sample fit when these predictors are added to the base model. Perhaps a significant improvement in predictive ability will be seen if we consider out of sample forecasts instead. To check this, we use a rolling window forecast for each model, beginning using the window t=1 through t=104 (the first two seasons) and forecasting the ILINet value at t=105, then shifting the window over by 1 and repeating the procedure of forecasting until we forecast the last value of ILINet available. We then compute the mean absolute error (MAE) for each model's set of forecasts. We find the MAE for the base model is 3.996544%, while that for the model utilizing google trends data is 4.254782%, representing an overall 6.461% *reduction* in fit.However, closer examination of the fit reveals a potentially salvageable result.

While overall predictive ability is certainly an interest, there are more specific aspects of this flu data that are of interest to public health officials. In particular, the values of ILINet around peak season (when ILINet is at its highest) are of special interest to public health officials, for these values relate to the most severe week of flu activity, and how severe it will be during this

time, which has consequences for advisory warnings issued to hospitals and citizens. We therefore have reason to consider more closely the periods of time around peak season for each season. We compute the MAE for each model using forecasts five weeks before, five weeks after, and at the peak week for the final 3 seasons captured in the data at hand. A rather encouraging picture then emerges. Two of the three seasons saw an improvement in fit, with one season experiencing a drastic improvement. More specifically, a 12.03294% improvement in MAE was seen in the first forecast season, a staggering 63.71317% improvement in MAE was seen in the second forecast season, and an unfortunate 13.26341% reduction in fit was seen in the third and final forecast season. (The model selected under the BIC criterion actually has a small improvement to MAE of approximately 1.8%. Additionally, it shows an marked improvement around season peaks, though these improvements are less pronounced.) We summarize these results in a graph of in sample fit (figure 10). Note the improvemnt of the Google Trends model around peak week for the first two seasons.

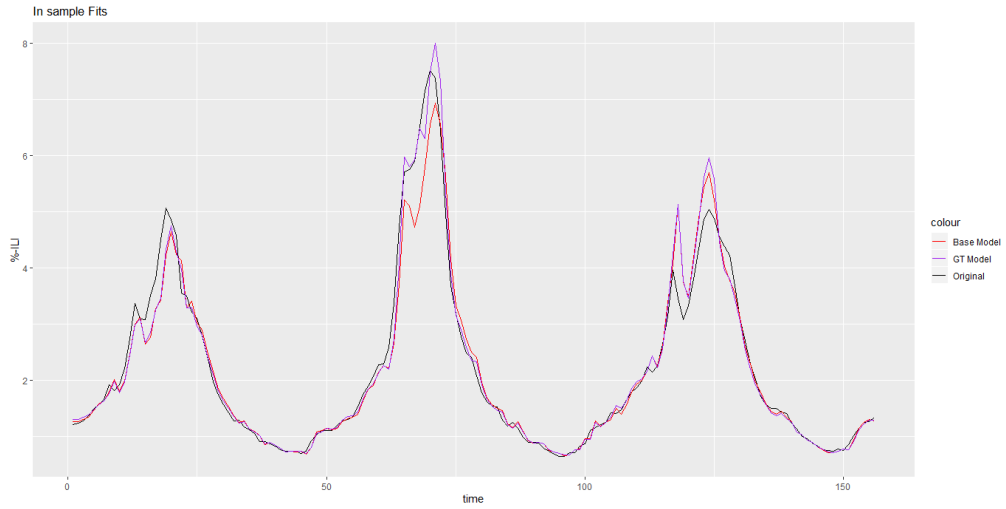

Figure 10: Final Comparison

It might be wondered what the results of trying to predict the current ILINet value using only google trends data would be. To this end, we consider building a model using search data for the query "flu." Since Google Trends broad matches search queries, the search for "flu" should encode most information related to flu-based searches. We run a best subset routine considering current values for the "flu" query search, as well as lag values of 1, 2 and 52. (See code provided in the appendix for details.) Using the AIC criterion, the model with all predictors is considered optimal. This model has a markedly decreased sample fit (adjusted $R^2 = 0.4768$). We then consider the same rolling window forecast considered above. The results are less than promising, with a MAE calculated as 0.254087, representing a staggering 535.7669% reduction in fit compared to the original baseline seasonal AR(2) model. We would therefore advise against using Google Trends data alone as a predictor in this case.

6 Conclusion

Two examples of how Google Trends data may be used to predict the present have been discussed. In one, we considered how Google Trends data might be used to forecast current values of economic indicators, such as car sales; in the other, we considered how Google Trends might be

used to forecast flu season. In the automobiles example, we found that both in sample fit, out of sample forecasting ability were improved by the inclusion of google trends data. In the flu season example, neither in sample fit, nor out of sample predictive ability were greatly improved. However, inclusion of Google Trends data did provide a significant improvement in predictive ability for certain specific aspects of the data, namely around each season's peak week. Similar improvements for specific aspects of the automobiles data, namely during the December 2007 through June 2009 recession, were also seen in the automobiles sales example when Google Trends data were considered.

These results demonstrate the potential usefulness of Google Trends in forecasting applications. If a base model doesn't have an extremely high in sample fit, adding Google search data could potentially improve that fit, and subsequent forecasting accuracy. Furthermore, even if in sample fit doesn't greatly benefit from the addition of Google Trends data, forecasting accuracy for specific aspects of the data of interest could still be improved.

We close with a suggestion for further research. It has been noted that combining sets of forecasts made with different models can be potentially more accurate (in terms of MSE) than individual forecasts made with a given model.[2] It is possible, then, that composite forecasting models can be built using several models built with google trends data to improve forecasting accuracy for different aspects of the data of interest (such as season peak for the flu data above) that yield overall increased accuracy for each aspect of the data considered. Thus, even greater accuracy can potentially be achieved using Google Trends data than seen in the examples presented here. We therefore encourage research be devoted to ascertaining whether this composite forecasting method could work as intuition would seem to suggest it might.

References

[1] Choi, H., & Varian, H. (2012). Predicting the present with Google Trends. Economic Record, 88, 2-9.

[2] Bates, J., & Granger, C. (1969). The Combination of Forecasts. OR, 20(4), 451-468.

Appendix A - Code for automobiles example

```
# base model
data1 = read.csv("/Users/lixuezhen/Desktop/237\ project/vehicles.csv",header = T,skip = 6)
head(data1)
y = log(data1$Value)
plot.ts(y[13:90],xlab = "year", xaxt = "n",main = "Motor Vehicles and Parts Dealers",ylab =
    "log(sales)")
axis(side = 1, at = c(13,25,37,49,61,73),labels = c(2006,2007,2008,2009,2010,2011))
acf(y[13:90],lag.max = 30,main = "ACF plot")
model1 = lm(y[13:90]~y[12:89]+y[1:78])
summary(model1)
library(leaps)
model1_2 = lm(y[13:90]~y[12:89]+y[11:88]+y[10:87]+y[9:86]+y[8:85]+y[1:78])
summary(model1_2)
fit_all = regsubsets(y[13:90]~y[12:89]+y[11:88]+y[10:87]+y[9:86]+y[8:85]+y[1:78],data = as.
    data.frame(y),nbest =1, method="exhaustive")
fit_all.display = as.data.frame(summary(fit_all)$outmat)
fit_all.display = cbind(fit_all.display,round(summary(fit_all)$rsq,4),round(summary(fit_all)
    $adjr2,4),round(summary(fit_all)$cp,4),round(summary(fit_all)$bic,4))
varnames = c("lag1","lag2","lag3","lag4","lag5","lag12")
names(fit_all.display)=c(varnames,"R^2","adj R^2","Cp","BIC")
fit_all.display

#diagnosis
par(mfrow = c(2,2))
plot(model1,which = 1:4)
par(mfrow = c(1,1))
acf(model1$residuals)
pacf(model1$residuals)


# google trends model
trucks_suvs  = read.csv("/Users/lixuezhen/Desktop/237\ project/trucks_suvs.csv",header = T,
    skip = 1)
auto_insurance = read.csv("/Users/lixuezhen/Desktop/237\ project/auto_insurance.csv",header
    = T,skip = 1)
head(trucks_suvs)
head(auto_insurance)
suvs = trucks_suvs$Geo..United.States
insurance = auto_insurance$Geo..United.States
par(mfrow = c(2,1))
plot.ts((y[13:90]-mean(y[13:90]))/sd(y[13:90]),xlab = "year", ylim = c(-2.2,2.2),xaxt = "n",
    main = "log(sales) vs trucks and suvs",ylab = "standardized series")
axis(side = 1, at = c(13,25,37,49,61,73),labels = c(2006,2007,2008,2009,2010,2011))
lines((suvs[13:90]-mean(suvs[13:90]))/sd(suvs[13:90]),lty = 2)
legend(61,2,legend = c("log(sales)","trucks and suvs"),lty = c(1,2),cex = 0.5)
plot.ts((y[13:90]-mean(y[13:90]))/sd(y[13:90]),xlab = "year", ylim = c(-2.2,2.2),xaxt = "n",
    main = "log(sales) vs auto insurance",ylab = "standardized series")
axis(side = 1, at = c(13,25,37,49,61,73),labels = c(2006,2007,2008,2009,2010,2011))
lines((insurance[13:90]-mean(insurance[13:90]))/sd(insurance[13:90]),lty = 2)
legend(61,2,legend = c("log(sales)","auto insurance"),lty = c(1,2),cex = 0.5)
par(mfrow = c(1,1))
model2 = lm(y[13:90]~y[12:89]+y[1:78]+suvs[13:90]+insurance[13:90])
summary(model2)

#diagnosis
par(mfrow = c(2,2))
plot(model2,which = 1:4)
par(mfrow = c(1,1))
par(mfrow = c(2,1))
acf(model2$residuals)
pacf(model2$residuals)
par(mfrow = c(1,1))

# final plots
```

```
plot.ts(y[13:90],ylab = "log(sales)",xlab = "year",xaxt = "n",col = "red",lty = 2,main = "
    Motor Vehicles and Parts")
axis(side = 1, at = c(13,25,37,49,61,73),labels = c(2006,2007,2008,2009,2010,2011))
lines(model1$fitted.values,lwd = 0.5,lty = 1)
lines(model2$fitted.values,lwd = 1,lty = 1)
legend(60,11.4,legend = c("actual","base","trend"),col = c("red","black","black"),lwd = c
    (1,0.5,1),lty = c(2,1,1))
legend(0,10.9,legend = c("MAE improvement","Overall = 10.44%","During recession = 31.79%"))

# MAE comparison
error1 = NULL
for (i in 31:90){
  fit =  lm(y[(i-18):(i-1)]~y[(i-19):(i-2)]+y[(i-30):(i-13)])
  error1 = c(error1,sum(fit$coefficients * c(1,y[i-1],y[i-12])) - y[i])
}
mae1 = sum(abs(error1))/length(error1)
mae1
error2 = NULL
for (i in 31:90){
  fit =  lm(y[(i-18):(i-1)]~y[(i-19):(i-2)]+y[(i-30):(i-13)]+suvs[(i-18):(i-1)]+insurance[(i
      -18):(i-1)])
  error2 = c(error2,sum(fit$coefficients * c(1,y[i-1],y[i-12],suvs[i],insurance[i])) - y[i])
}

mae2 = sum(abs(error2))/length(error2)
mae2
(mae1-mae2)/mae1 # overall improvement

mae1 = sum(abs(error1[18:35]))/length(error1[18:35])
mae2 = sum(abs(error2[18:35]))/length(error2[18:35])
(mae1-mae2)/mae1 # recession period improvement

# leave out the the end of the actual data set
predict_value = NULL
for (i in 18:90){
fit = lm(y[13:(i-1)]~y[12:(i-2)]+y[1:(i-13)]+suvs[13:(i-1)]+insurance[13:(i-1)])
predict_value = c(predict_value,sum(fit$coefficients * c(1,y[i-1],y[i-12],suvs[i],insurance[
    i])))
}
plot.ts(y[18:90],xlab = "year",ylab= "log(sales)",xaxt = "n",main = "Prediction for sales
    with google trends",ylim = c(10.7,11.45))
axis(side = 1, at = c(8,20,32,44,56,68),labels = c(2006,2007,2008,2009,2010,2011))
lines(predict_value,lty = 2)
legend(60,11.4,legend = c("actual","prediction"),lty = c(1,2))

mean(abs(predict_value-y[18:90]))
```

Appendix B - Code for flu data example

```
library(ggplot2)
library(leaps)
library(MASS)
setwd("C:/Users/Kenneth/Downloads/Flu (15.16-18.19)")
flu<-read.csv("ILINetNational.csv", header=TRUE)

n<-length(flu$X.WEIGHTED.ILI)
p1<-ggplot(data=flu, aes(x=1:n,y=X.WEIGHTED.ILI))+geom_line()+geom_point()
p1+labs(title="Weighted ILI", y="%-ILI", x="week")
x11()
Xt<-1/flu$X.WEIGHTED.ILI
p2<-ggplot(data=as.data.frame(Xt), aes(x=1:n,y=Xt))+geom_line()+geom_point()
p2+labs(title="Inverse ILI", y="1/%-ILI", x="week")
model1<-lm(flu$X.WEIGHTED.ILI~L1+L2+L52)
boxcox(model1)##This suggests an inverse transformation, which we use below.

##ACF and PACF plots; suggestive of an AR(2) model.
```

```
par(mfrow=c(1,2))
acf(Xt,lag=53)
pacf(Xt,lag=53)

##Model fit with Regression approach seen in the paper:
L1<-c(NA,Xt[1:(NROW(Xt)-1)])
L2<-c(NA,NA,Xt[1:(NROW(Xt)-2)])
L52<-c(rep(NA,52),Xt[1:(NROW(Xt)-52)])

model1<-lm(Xt~L1+L2+L52)
summary(model1)
par(mfrow=c(2,2))
plot(model1, which=1)
plot(model1, which=2)
plot(model1, which=3)
plot(model1, which=5)

np<-length(model1$fitted)
ggplot() +geom_line(data=as.data.frame(model1$fitted), aes(x=1:np,y=model1$fitted), color='
    red')+
geom_line(data=as.data.frame(Xt[(n+1-np):n]), aes(x=1:np,y=Xt[(n+1-np):n]), color='black')

##Once a base model has been chosen, we can use GT data as additional predictors:
fluseason<-read.csv("fluseasontrends.csv", header=TRUE)
flushot<-read.csv("flushottrends.csv", header=TRUE)
flusymptoms<-read.csv("flusymptomstends.csv", header=TRUE)
flutrends<-read.csv("flutrends.csv", header=TRUE)

DATA<-as.data.frame(cbind(Xt,L1,L2,L52,fluseason,flushot,flusymptoms,flutrends))
best<-regsubsets(Xt~L1+L2+L52+flu.season+flu.shot+flu.symptoms+flu, data=DATA)
ss1<-summary(best)
n1<-length(Xt)
p1<-as.integer(rownames(ss1$which))+1
aicp1<-n1*log(ss1$rss/n1)+2*p1
object1<-cbind(ss1$which, ss1$rss, ss1$rsq, ss1$adjr2, ss1$cp, aicp1, ss1$bic)
colnames(object1)<-c(colnames(ss1$which), "sse", "Rsq", "adjRsq", "Cp", "AIC", "BIC")
object1

model2<-lm(Xt~L1+L2+L52+fluseason$flu.season+flusymptoms$flu.symptoms+flutrends$flu)
##This appears to be the best model using utilizing the GT data AIC Criterion
summary(model2)
##Rolling forecasts:
##Modle1 Forecasts; N=n-m subsamples
m<-104
error<-NULL
for(i in 1:m){
XR<-Xt[i:(i+m-1)]
lag1<-c(NA,XR[1:(NROW(XR)-1)])
lag2<-c(NA,NA,XR[1:(NROW(XR)-2)])
lag52<-c(rep(NA,52),XR[1:(NROW(XR)-52)])
fit<-lm(XR~lag1+lag2+lag52)
error[i]<-(sum(fit$coefficients*c(1,Xt[i+m-1],Xt[i+m-2],Xt[i+m-52])))-Xt[i+m]
}
MAE1<-sum(abs(error))/length(error)
MAE1

##Model2 rolling forecasts
error2<-NULL
for(i in 1:m){
XR<-Xt[i:(i+m-1)]
lag1<-c(NA,XR[1:(NROW(XR)-1)])
lag2<-c(NA,NA,XR[1:(NROW(XR)-2)])
lag52<-c(rep(NA,52),XR[1:(NROW(XR)-52)])
trend1<-fluseason$flu.season[i:(i+m-1)]
trend2<-flusymptoms$flu.symptoms[i:(i+m-1)]
trend3<-flutrends$flu[i:(i+m-1)]
fit<-lm(XR~lag1+lag2+lag52+trend1+trend2+trend3, data=NULL)
```

```
error2[i]<-(sum(fit$coefficients*c(1,Xt[i+m-1],Xt[i+m-2],Xt[i+m-52],fluseason$flu.season[i+m
    ],flusymptoms$flu.symptoms[i+m],flutrends$flu[i+m])))-Xt[i+m]
}
MAE2<-sum(abs(error2))/length(error2)
MAE2

(MAE1-MAE2)/MAE1

##MAE Comparison around peaks
m1<-max(flu$X.WEIGHTED.ILI[53:104])
m2<-max(flu$X.WEIGHTED.ILI[105:156])
m3<-max(flu$X.WEIGHTED.ILI[157:208])

flu$X.WEIGHTED.ILI==m1
flu$X.WEIGHTED.ILI==m2
flu$X.WEIGHTED.ILI==m3
##The last three peaks occur at points 71, 122, and 176

error1b<-NULL
error1g<-NULL
error2b<-NULL
error2g<-NULL
error3b<-NULL
error3g<-NULL
for(i in -5:5){
error1b[i+6]<-model1$fitted[i+71-52]-Xt[i+71]
error1g[i+6]<-model2$fitted[i+71-52]-Xt[i+71]
error2b[i+6]<-model1$fitted[i+122-52]-Xt[i+122]
error2g[i+6]<-model2$fitted[i+122-52]-Xt[i+122]
error3b[i+6]<-model1$fitted[i+176-52]-Xt[i+176]
error3g[i+6]<-model2$fitted[i+176-52]-Xt[i+176]
}
MAEb1<-sum(abs(error1b))/length(error1b)
MAEg1<-sum(abs(error1g))/length(error1g)
MAEb2<-sum(abs(error2b))/length(error2b)
MAEg2<-sum(abs(error2g))/length(error2g)
MAEb3<-sum(abs(error3b))/length(error2b)
MAEg3<-sum(abs(error3g))/length(error3g)

(MAEb1-MAEg1)/MAEb1
(MAEb2-MAEg2)/MAEb2
(MAEb3-MAEg3)/MAEb3

np<-length(model1$fitted)
##Model in-sample fit comparisons
plot<-ggplot()+geom_line(data=as.data.frame(Xt[(n+1-np):n]), aes(x=1:np,y=1/Xt[(n+1-np):n],
    color="Original"))+
geom_line(data=as.data.frame(model1$fitted), aes(x=1:np,y=1/model1$fitted,color="Base Model"
    ))+
geom_line(data=as.data.frame(model2$fitted), aes(x=1:np,y=1/model2$fitted, color="GT Model")
    )
plot+labs(title="In sample Fits", y="%-ILI", x="time")+
scale_color_manual(values=c("red", "purple", "black"))


##Predict flu using flu shot GT data only:
St<-flutrends$flu
ggplot()+geom_line(data=flu, aes(x=1:n,y=X.WEIGHTED.ILI), color='black')+
geom_line(data=as.data.frame(St), aes(x=1:n,y=St), color='red')

gl1<-c(NA,St[1:(NROW(St)-1)])
gl2<-c(NA,NA,St[1:(NROW(St)-2)])
gl52<-c(rep(NA,52),St[1:(NROW(St)-52)])
subset<-regsubsets(Xt~St+gl1+gl2+gl52,data=NULL)
ss2<-summary(subset)
n2<-length(St)
p2<-as.integer(rownames(ss2$which))+1
aicp2<-n*log(ss2$rss/n2)+2*p2
```

```
object2<-cbind(ss2$which, ss2$rss, ss2$rsq, ss2$adjr2, ss2$cp, aicp2, ss2$bic)
colnames(object2)<-c("int","lag0","lag1","lag2","lag52", "sse", "Rsq", "adjRsq", "Cp", "AIC", "BIC")
object2

##Model with all predictors apperas to be the best (by AIC criterion)
##GT model rolling forecasts
error3<-NULL
for(i in 1:m){
GTR<-St[i:(i+m-1)]
XR<-Xt[i:(i+m-1)]
lag1<-c(NA,GTR[1:(NROW(GTR)-1)])
lag2<-c(NA,NA,GTR[1:(NROW(GTR)-2)])
lag52<-c(rep(NA,52),GTR[1:(NROW(GTR)-52)])
fit<-lm(XR~GTR+lag1+lag2+lag52, data=NULL)
error3[i]<-(sum(fit$coefficients*c(1,St[i+m],St[i+m-1],St[i+m-2],St[i+m-52])))-Xt[i+m]
}
MAE3<-sum(abs(error3))/length(error3)
MAE3

(MAE1-MAE3)/MAE1 ##Note the extreme reduction in fit, reflected in the graph below.

model3<-lm(Xt~St+gl1+gl2+gl52, data=NULL)
np2<-length(model3$fitted)
##Model in sample fit comparisons
plot2<-ggplot()+geom_line(data=as.data.frame(Xt[(n+1-np2):n]), aes(x=1:np2,y=1/Xt[(n+1-np2):n],color="Original"))+
geom_line(data=as.data.frame(model1$fitted), aes(x=1:np,y=1/model1$fitted,color="Base Model"))+
geom_line(data=as.data.frame(model3$fitted), aes(x=1:np2,y=1/model3$fitted, color="GT Pure Model"))

plot2+labs(title="In sample Fits", y="%-ILI", x="time")+
scale_color_manual(values=c("red", "purple", "black"))

##----------------------------
##The following code may be used to run the analysis discussed in the Flu example
##using the google trends model selected by the BIC criterion.
model2<-lm(Xt~L1+L2+L52+flushot$flu.shot)
##This appears to be the best model using utilizing the GT data BIC criterion
summary(model2)
##Rolling forecasts:
##Modle1 Forecasts; N=n-m subsamples
m<-104
error<-NULL
for(i in 1:m){
XR<-Xt[i:(i+m-1)]
lag1<-c(NA,XR[1:(NROW(XR)-1)])
lag2<-c(NA,NA,XR[1:(NROW(XR)-2)])
lag52<-c(rep(NA,52),XR[1:(NROW(XR)-52)])
fit<-lm(XR~lag1+lag2+lag52)
error[i]<-(sum(fit$coefficients*c(1,Xt[i+m-1],Xt[i+m-2],Xt[i+m-52])))-Xt[i+m]
}
MAE1<-sum(abs(error))/length(error)
MAE1

##Model2 rolling forecasts
error2<-NULL
for(i in 1:m){
XR<-Xt[i:(i+m-1)]
lag1<-c(NA,XR[1:(NROW(XR)-1)])
lag2<-c(NA,NA,XR[1:(NROW(XR)-2)])
lag52<-c(rep(NA,52),XR[1:(NROW(XR)-52)])
trend1<-flushot$flu.shot[i:(i+m-1)]
fit<-lm(XR~lag1+lag2+lag52+trend1, data=NULL)
error2[i]<-(sum(fit$coefficients*c(1,Xt[i+m-1],Xt[i+m-2],Xt[i+m-52],flushot$flu.shot[i+m])))-Xt[i+m]}
MAE2<-sum(abs(error2))/length(error2)
```

```
MAE2

(MAE1-MAE2)/MAE1

##MAE Comparison around peaks
m1<-max(flu$X.WEIGHTED.ILI[53:104])
m2<-max(flu$X.WEIGHTED.ILI[105:156])
m3<-max(flu$X.WEIGHTED.ILI[157:208])

flu$X.WEIGHTED.ILI==m1
flu$X.WEIGHTED.ILI==m2
flu$X.WEIGHTED.ILI==m3
##The last three peaks occur at points 71, 122, and 176

error1b<-NULL
error1g<-NULL
error2b<-NULL
error2g<-NULL
error3b<-NULL
error3g<-NULL
for(i in -5:5){
error1b[i+6]<-model1$fitted[i+71-52]-Xt[i+71]
error1g[i+6]<-model2$fitted[i+71-52]-Xt[i+71]
error2b[i+6]<-model1$fitted[i+122-52]-Xt[i+122]
error2g[i+6]<-model2$fitted[i+122-52]-Xt[i+122]
error3b[i+6]<-model1$fitted[i+176-52]-Xt[i+176]
error3g[i+6]<-model2$fitted[i+176-52]-Xt[i+176]
}
MAEb1<-sum(abs(error1b))/length(error1b)
MAEg1<-sum(abs(error1g))/length(error1g)
MAEb2<-sum(abs(error2b))/length(error2b)
MAEg2<-sum(abs(error2g))/length(error2g)
MAEb3<-sum(abs(error3b))/length(error2b)
MAEg3<-sum(abs(error3g))/length(error3g)

(MAEb1-MAEg1)/MAEb1
(MAEb2-MAEg2)/MAEb2
(MAEb3-MAEg3)/MAEb3
```