# Demography Predictive Modeling Working Group - What is predictive modeling?

*Corey Sparks, PhD*

*10/8/2019*

## Where are we going?

This working group was formed to learn more about predictive modeling and learning how to correctly use it in demographic scenarios. Not only are these skills valuable, but in the data science world they are almost assumed. Moreover, the tradition social science methodological toolkit lacks these methods entirely, so here we are.

We are going to be exploring the various aspects of predictive modeling over the next few months, along the way we will see models we've seen before and those that we haven't. We'll also see things about model development that seem strange and alien to our social science sensibilities

### Setup

- For now we will use R
- R & Rstudio
  - caret package
- Later we will move into using Python's libraries
  - Need to install Anaconda Python distribution:
  - Anaconda
    - conda -
    - create -n myenv python=3.6
    - activate myenv
    - pip install sklearn tensorflow

## Inferential modeling

- Inferential modeling is when we use statistical models to ask questions about data
- We are generally interested in testing some set of ideas, and if they are supported by data

## What is predictive modeling?

- In predictive modeling we are more interested in developing a model or suite of models that accurately predict data.
- We are most concerned with a model that predicts unobserved data correctly.

## Some vocabulary

Predictive modeling has some vocabulary that we need to get around first, because again, like in everything else, people have completely re-invented the terminology to suit them.

Some of these are stolen from here


Figure 1: dangerous mission

**General terms**

- Machine learning - Mitchell (1997) provides a succinct definition: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T , as measured by P, improves with experience E.In simple language machine learning is a field in which human made algorithms have an ability learn by itself or predict future for unseen data. ML automates analytical model building. It uses methods from neural networks, statistics, operations research and physics to find hidden insights in data without being explicitly programmed where to look or what to conclude.
- Neural network - A neural network is a kind of machine learning inspired by the workings of the human brain. It's a computing system made up of interconnected units (like neurons) that processes information by responding to external inputs, relaying information between each unit. The process requires multiple passes at the data to find connections and derive meaning from undefined data.
- Deep learning - Deep Learning is derived from one machine learning algorithm called perceptron or multi layer perceptron that gain more and more attention nowadays because of its success in different fields like, computer vision to signal processing and medical diagnosis to self-driving cars. As all other AI algorithms deep learning is from decades, but now today we have more and more data and cheap computing power that make this algorithm really powerful to achieve state of the art accuracy. In the modern world this algorithm known as **artificial neural network**. Deep learning uses huge neural networks with many layers of processing units, taking advantage of advances in computing power and improved training techniques to learn complex patterns in large amounts of data. Common applications include image and speech recognition.
- Artificial intelligence - The word Artificial Intelligence comprises of two words "Artificial" and "Intelligence". Artificial refers to something which is made by human or non natural thing and Intelligence means ability to understand or think. There is a misconception that Artificial Intelligence is a system, but it is not a system. AI is implemented in the system. In other words, it's a model that's created by people then applied by computers to make decisions about things. "It is the study of how to train the computers so that computers can do things which at present human can do better."Therefore It is a intelligence where we want to add all the capabilities to machine that human contain.
- Bias = how wrong you are about predicting/estimating something. Low bias means you are very good at predicting something, high bias is the opposite
- Variance = How tightly packed are your predictions for a particular observation relative to each other? **Low variance** suggests your model is internally consistent, with predictions varying little from each other after every iteration. **High variance** (with low bias) suggests your model may be over-fitting and reading too deeply into the noise found in every training set.
- Instance = Observation
- Variables = Features - Think predictors in a model
  - Feature selection = variable selection - You have lots of variables measured, which ones are you going to use for a given model? This can be automated, random or based on expertise
- Algorithm = Think model, in predictive modeling, there are lots and lots of algorithms that we can use
- Parameters = unknown parts of a model that we need to estimate. These are the $\beta$'s in a regression model, or even the mean of a distribution

**Learning**

- Unsupervised Learning = Training a model to find patterns in an unlabeled dataset (e.g. clustering). i.e. you don't know which group an observation belongs to, but you want to try to group similar observations into groups
- Supervised learning = Training a model using a labeled dataset. i.e. You know which group an observation belongs to, and you want to try to make a model that predicts this correctly
- Training Set - A set of observations used to generate machine learning models.

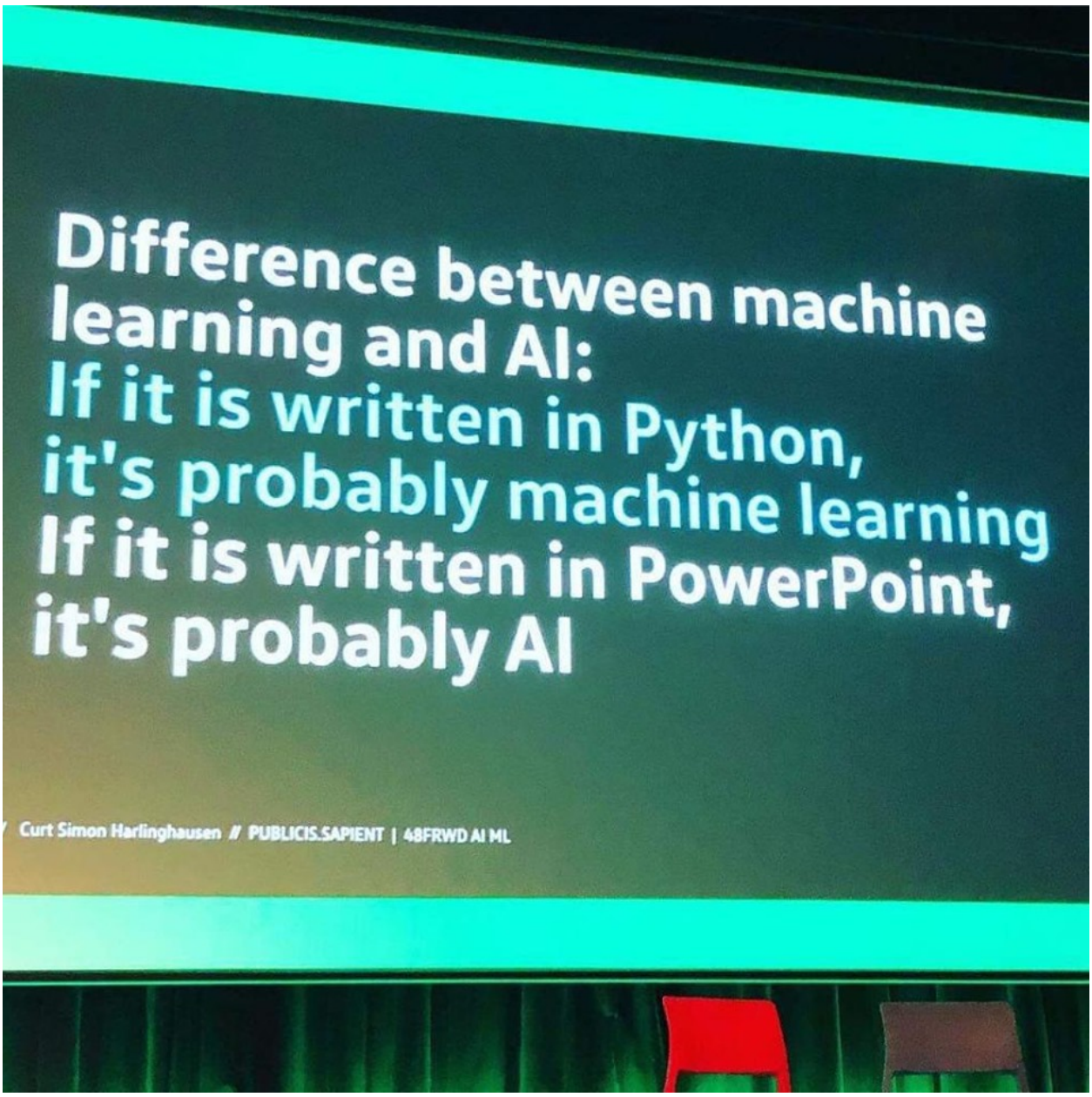



Figure 2: cartoon

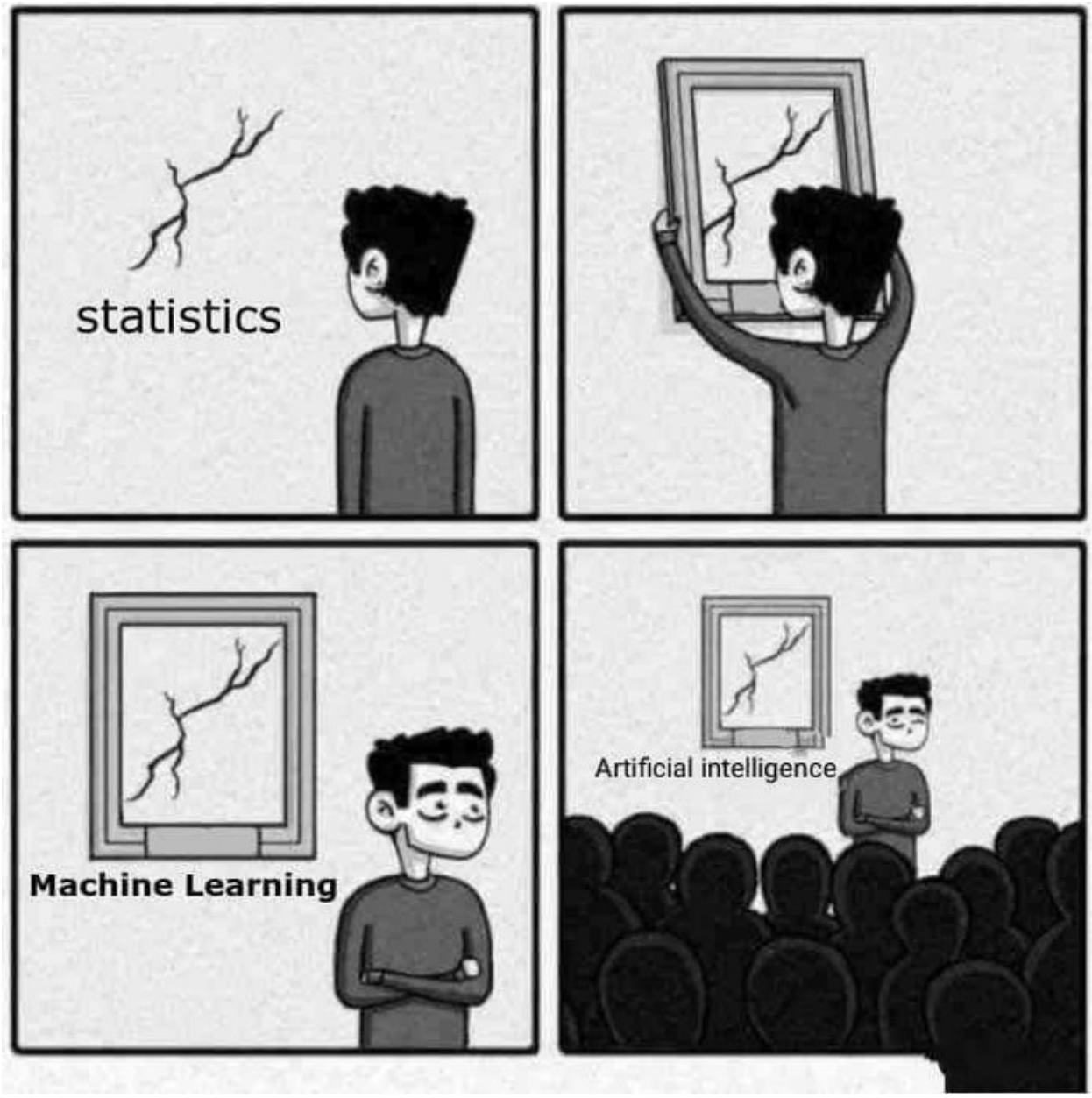



Figure 3: cartoon

- Test set - A set of observations used at the end of model training and validation to assess the predictive power of your model. How generalizable is your model to unseen data?

**Modeling terms**

- Classification = Predicting a categorical output (e.g. yes or no?, blue, green or red?).
  - Accuracy = Percentage of correct predictions made by the model.
- Clustering= Unsupervised grouping of data into buckets.
  - buckets = groups
- Hyperparameters/Tuning parameters - Secondary parameters in some models that control how the model works.
- Loss - Loss = true_value(from data-set)- predicted value(from ML-model) The lower the loss, the better a model (unless the model has over-fitted to the training data). The loss is calculated on training and validation and its interpretation is how well the model is doing for these two sets. Unlike accuracy, loss is not a percentage. It is a summation of the errors made for each example in training or validation sets.
- Cross-validation = A mechanism for estimating how well a model will generalize to new data by testing the model against one or more non-overlapping data subsets withheld from the training set.
- Over fitting - Over-fitting occurs when your model learns the training data too well and incorporates details and noise specific to your dataset. You can tell a model is over-fitting when it performs great on your training/validation set, but poorly on your test set (or new real-world data).
- Under fitting - Under-fitting occurs when your model over-generalizes and fails to incorporate relevant variations in your data that would give your model more predictive power. You can tell a model is under-fitting when it performs poorly on both training and test sets.
- Confusion matrix = Table that describes the performance of a classification model by grouping predictions into 4 categories.
  - True Positives: we correctly predicted they do have diabetes
  - True Negatives: we correctly predicted they don't have diabetes
  - False Positives: we incorrectly predicted they do have diabetes (Type I error)
  - False Negatives: we incorrectly predicted they don't have diabetes (Type II error)

## Training a model

When we train a model, we are fitting it to the training set of our data, typically 75-80% of the data.

```
library(readr)
prb<-read_csv(file = "https://raw.githubusercontent.com/coreysparks/data/master/PRB2008_All.csv", col_ty
names(prb)<-tolower(names(prb))

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
```

```
prb<-prb%>%
  mutate(africa=ifelse(continent=="Africa", 1, 0))%>%
  filter(complete.cases(imr, tfr, continent,percpoplt15, e0total, percurban, percmarwomcontramodern))%>%
  select(imr, tfr, continent,africa,percpoplt15, e0total, percurban, percmarwomcontramodern)
```

```
knitr::kable(head(prb))
```

| imr | tfr | continent | africa | percpoplt15 | e0total | percurban | percmarwomcontramodern |
|---|---|---|---|---|---|---|---|
| 163.0 | 6.8 | Asia | 0 | 45 | 43 | 20 | 9 |
| 8.0 | 1.6 | Europe | 0 | 27 | 75 | 45 | 8 |
| 27.0 | 2.3 | Africa | 1 | 30 | 72 | 63 | 52 |
| 132.0 | 6.8 | Africa | 1 | 46 | 43 | 57 | 5 |
| 26.0 | 1.7 | Asia | 0 | 21 | 71 | 64 | 20 |
| 4.7 | 1.9 | Oceania | 0 | 19 | 81 | 87 | 75 |

### Create data partition

Here we use 80% of the data to train our simple model

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
set.seed(1115)
train<- createDataPartition(y = prb$imr , p = .80, list=F)

prbtrain<-prb[train,]
prbtest<-prb[-train,]
```

```
lmcontrol<-trainControl(method="none", number=1)

lmt<-train(tfr~e0total, data=prbtrain, method="lm", trControl=lmcontrol )
lmt
```

```
## Linear Regression
##
## 135 samples
##   1 predictor
##
## No pre-processing
## Resampling: None
```

```
summary(lmt)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -3.6099 -0.5597  0.0786  0.6255  2.6549
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.486527   0.542867   21.16   <2e-16 ***
## e0total     -0.123535   0.008051  -15.34   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.007 on 133 degrees of freedom
## Multiple R-squared:  0.639,  Adjusted R-squared:  0.6363
## F-statistic: 235.4 on 1 and 133 DF,  p-value: < 2.2e-16
```

```
prbtrain%>%
  ggplot(aes(x=e0total, y=tfr))+
  geom_point()+
  geom_smooth(method = "lm", se = FALSE)+
  ggtitle(label="PRB data training set linear model")
```


We can get a numeric summary of how well the model fit the data here. This for a linear model gives us the Root Mean squared error (RMSE), model R square and the Mean Absolute Error (MAE).

$\text{RMSE} = \sqrt{(\text{OBSERVED}_i - \text{PREDICTED}_i)^2}$

This is one example of a **Loss function**

$\text{R}^2 = 1 - \frac{SSR}{SST}$

$\text{MAE} = \frac{\sum_{i=1}^{n} |\text{OBSERVED}_i - \text{PREDICTED}_i|}{n}$ is another loss function

We can summarize the model fit to the training data here:

```
prbtrain$pred<-fitted.values(lmt)
prbtrain$obs<-prbtrain$tfr

obsdat<-data.frame(obs=prbtrain$obs, pred=prbtrain$pred)
defaultSummary(data=obsdat)
```

```
##      RMSE  Rsquared       MAE
## 0.9991744 0.6390079 0.7607771
```

## Testing a model

Here we use the test data to evaluate how well the model performs on data that were not used to estimate the model's parameters

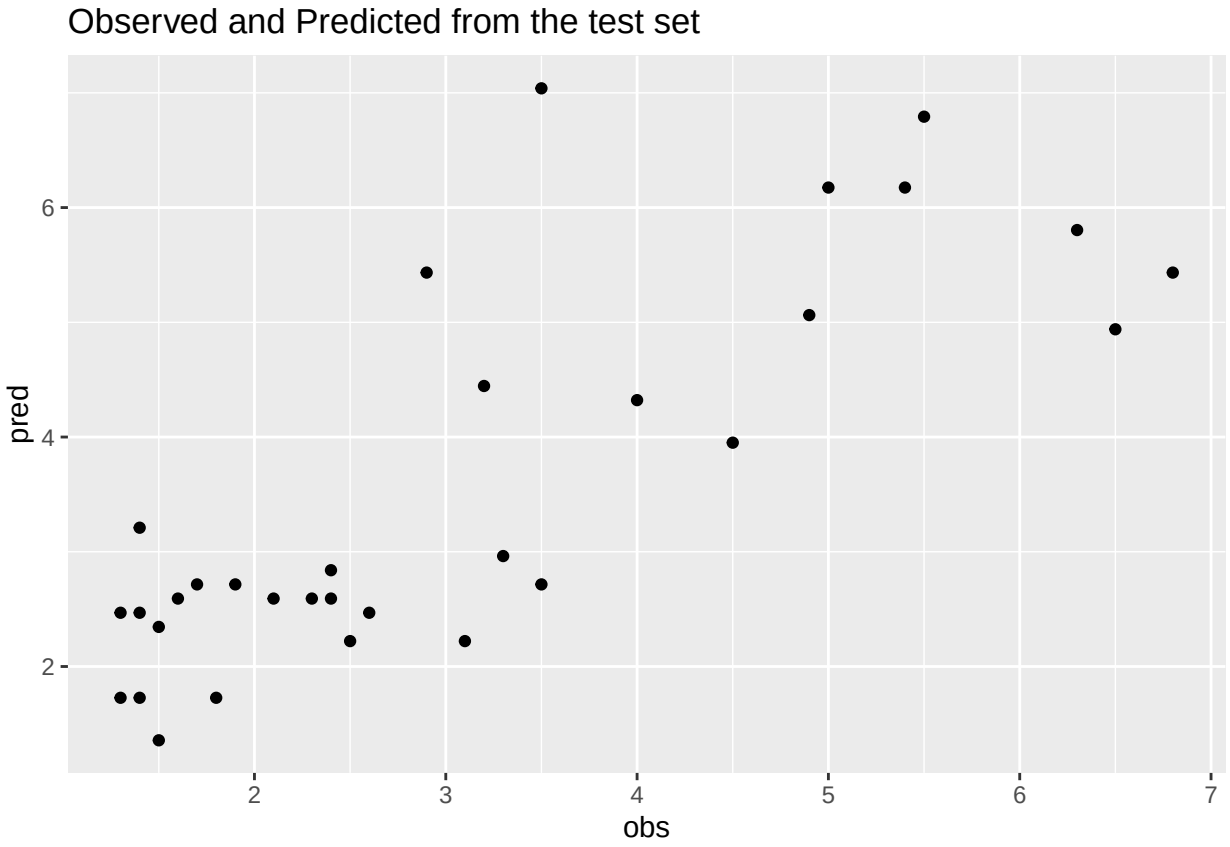
```
#get the predicted values from the model
prbtest$pred<-predict(lmt, newdata=prbtest)

modelvalues<-data.frame(obs = prbtest$tfr, pred=prbtest$pred)
head(modelvalues)
```

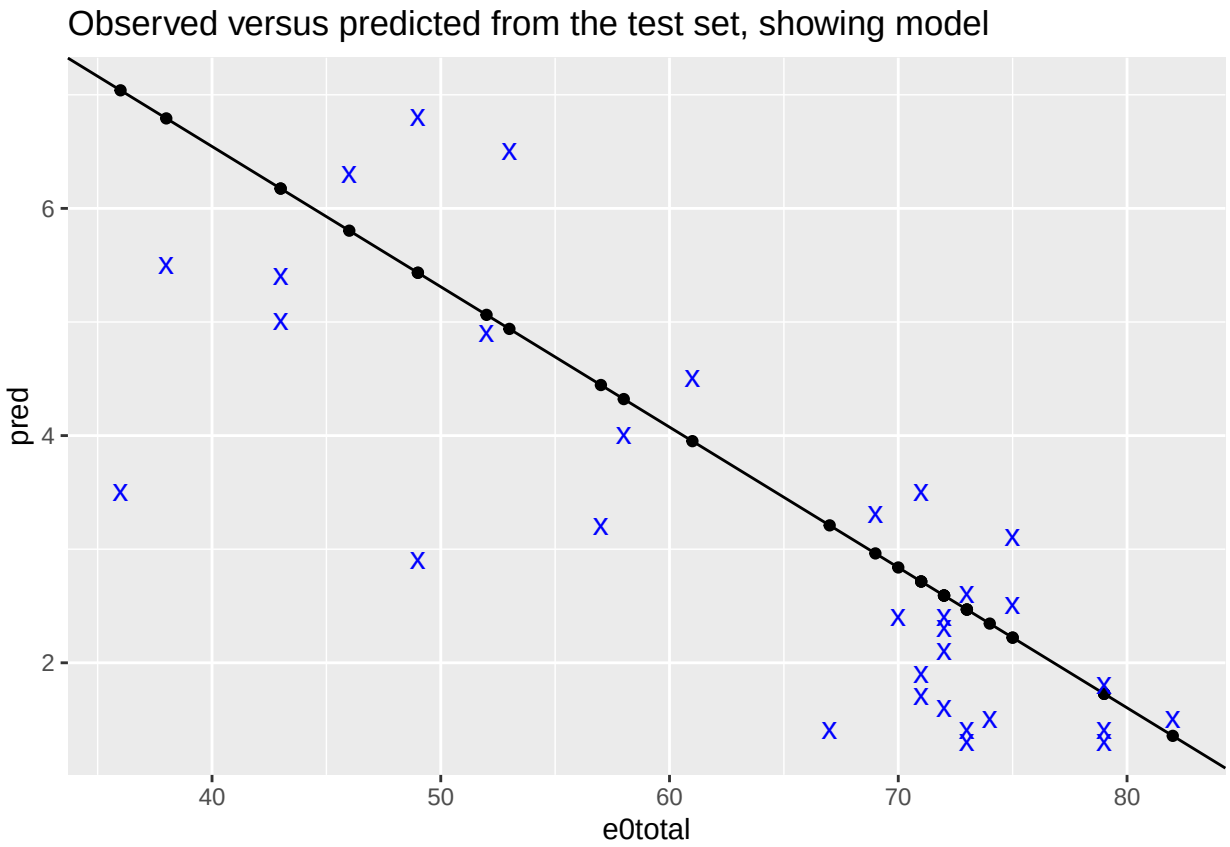
```
##   obs     pred
## 1 2.3 2.592042
## 2 1.7 2.715577
## 3 2.5 2.221438
## 4 2.9 5.433336
## 5 1.4 2.468508
## 6 6.8 5.433336
```

```
#Linear parameters from the regression
int<- coef(lmt$finalModel)[1]
slop<- coef(lmt$finalModel)[2]

modelvalues%>%
  ggplot(aes(x=obs, y=pred))+
  geom_point()+
  ggtitle(label = "Observed and Predicted from the test set")
```




```
prbtest%>%
  ggplot(aes(x=e0total, y=pred))+
  geom_point()+
  geom_abline(intercept =int, slope=slop)+
  geom_point(aes(x=e0total, y=tfr), color="blue", pch="x", cex=4) +
  ggtitle("Observed versus predicted from the test set, showing model")
```




Here we get out numeric summary of the model fit on the test set

```
defaultSummary(data=obsdat)
```

```
##      RMSE  Rsquared       MAE
## 0.9991744 0.6390079 0.7607771
```

```
defaultSummary(data=modelvalues)
```

```
##      RMSE  Rsquared       MAE
## 1.1287502 0.6402727 0.8602180
```

so the model fit the training data better than the test data, according to RMSE and $R^2$, but the MAE was lower for the test set.

## Changing the model

Now, let's add more stuff to the model! We will also use PCA to pre-process the predictors

```
lmt2<-train(tfr~e0total+africa+percpoplt15+ percmarwomcontramodern, data=prbtrain, method="lm", trContr
lmt2
```

```
## Linear Regression
##
## 135 samples
##   4 predictor
##
## Pre-processing: principal component signal extraction (4), centered
```

```
##  (4), scaled (4)
## Resampling: None
```

```
summary(lmt2)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.98483 -0.44770  0.07453  0.37093  1.71476
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.26370    0.05427  60.138  < 2e-16 ***
## PC1         -0.88119    0.03185 -27.664  < 2e-16 ***
## PC2         -0.19194    0.07826  -2.453   0.0155 *
## PC3          0.52584    0.08685   6.055 1.39e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6306 on 131 degrees of freedom
## Multiple R-squared:  0.8605, Adjusted R-squared:  0.8573
## F-statistic: 269.3 on 3 and 131 DF,  p-value: < 2.2e-16
```

```
prbtrain$pred<-fitted.values(lmt2)
prbtrain$obs<-prbtrain$tfr

obsdat<-data.frame(obs=prbtrain$obs, pred=prbtrain$pred)



prbtest$pred<-predict(lmt2, newdata=prbtest)

modelvalues<-data.frame(obs = prbtest$tfr, pred=prbtest$pred)

defaultSummary(data=obsdat)
```

```
##      RMSE  Rsquared       MAE
## 0.6211499 0.8604892 0.4910527
```

```
defaultSummary(data=modelvalues)
```

```
##      RMSE  Rsquared       MAE
## 0.7573865 0.8217829 0.5708392
```

So we could say this model is more **accurate** than the first model

### Try a different model

Here we try a recursive partition decision tree model, or RPART for short

```
#rpcontrol<-trainControl(method="none", number =1)
#cpgrid<-expand.grid(.cp = seq(.0001, .05, length.out = 20))
```

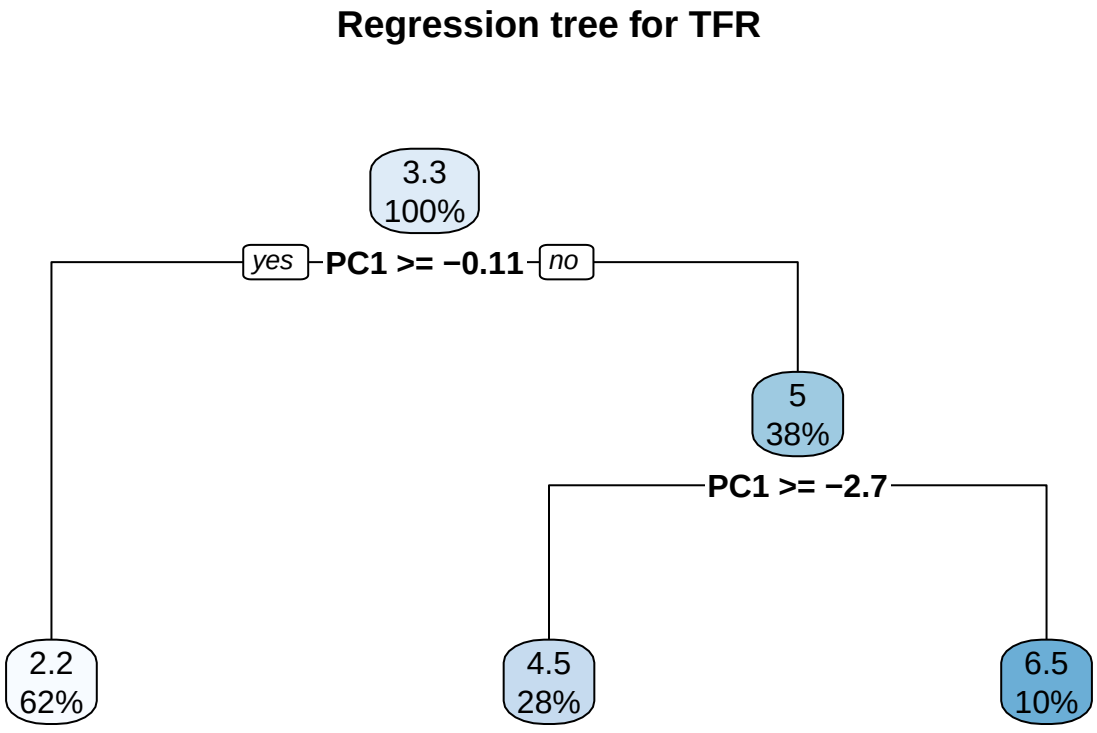
```
rpm1<-train(tfr~e0total+africa+percpoplt15+ percmarwomcontramodern, data=prbtrain, method="rpart",  prel
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
## trainInfo, : There were missing values in resampled performance measures.
```

```
rpm1
```

```
## CART
##
## 135 samples
##   4 predictor
##
## Pre-processing: principal component signal extraction (4), centered
##  (4), scaled (4)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 135, 135, 135, 135, 135, 135, ...
## Resampling results across tuning parameters:
##
##   cp          RMSE       Rsquared   MAE
##   0.08809828  0.9322673  0.7043617  0.7410548
##   0.09959688  0.9510389  0.6926181  0.7611599
##   0.70857991  1.6229779  0.5347022  1.3582322
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was cp = 0.08809828.
```

```
rpart.plot::rpart.plot(rpm1$finalModel, main="Regression tree for TFR")
```



```
prbtrain$pred<-fitted.values(rpm1)
prbtrain$obs<-prbtrain$tfr

obsdat<-data.frame(obs=prbtrain$obs, pred=prbtrain$pred)


prbtest$pred<-predict(rpm1, newdata=prbtest)

modelvalues<-data.frame(obs = prbtest$tfr, pred=prbtest$pred)

defaultSummary(data=obsdat)
```

```
##      RMSE  Rsquared       MAE
## 0.8034401 0.7665886 0.6490828
```

```
defaultSummary(data=modelvalues)
```

```
##      RMSE  Rsquared       MAE
## 0.9898982 0.7383476 0.8205634
```

Here we see the linear model with PCA processing did better in terms of prediction than the regression tree.