Model data

Isaac Verminck

2018-04-06

```
load(file = paste(data_path, "year_of_birth.RData", sep = "/"))
```

Model

Variables

In the previous sections we looked at 3 variables which could help predict the number of marriages:

- region
- year
- age

Visually we saw region and year have nearly no impact on the number of marriages. Age was the main driver. Let's see if we can quantify this relation between age and number of marriages.

Data

In order to have a serious analysis we have to divide our data in both a training set and a test set. The training set we use for our visualizations and model fitting. Data in the test set is used to test our models. We want to avoid overoptimistic models. Two thirds of the data is used for training, one third is used for testing.

```
marriage <- year_of_birth %>% # most modelling takes place on the aggregate level of belgium
  filter(type == "marriage" & region == "belgium")  %>% # so we don't mention belgium in the object nam
  age_continuous
```

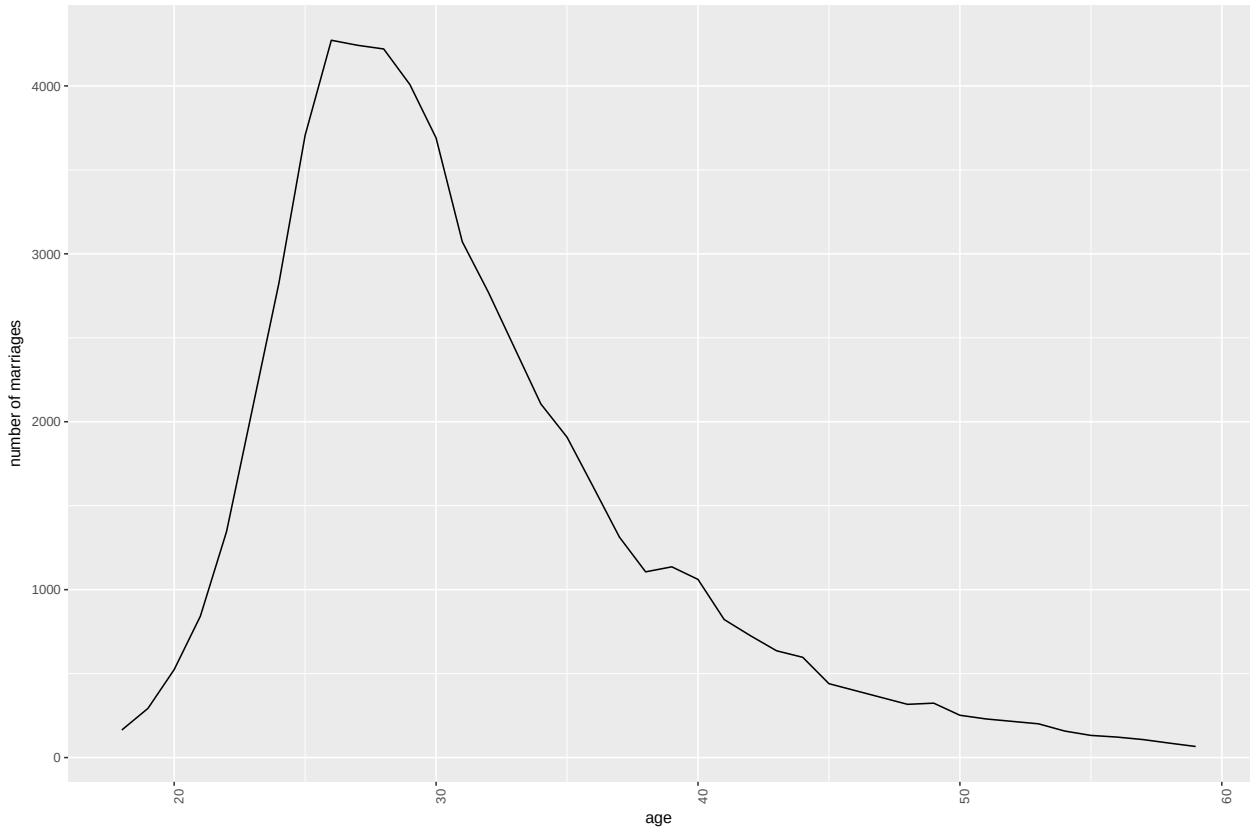
```
ind <- 1:nrow(marriage)
train_ind <- sample(ind,
                    size = length(ind)*2/3)
test_ind <- setdiff(ind, train_ind)

marriage_train <- slice(marriage, train_ind)
marriage_test <- slice(marriage, test_ind)
```

Choice of model

If you recall well, the shape of the distribution was very similar to a parabole. In the previous visualization we only focused on the most recent data (2014), here we'll take the data from 2013 into account as well. To do this we take the average of both years.

```
# average both years
marriage_train <- marriage_train %>%
  group_by(age, region) %>%
  summarise(nr_marriages = round(mean(quantity),
                                 digits = 0))
```

```r
# graph
ggplot(data = marriage_train,
       aes(x = age, y = nr_marriages)) +
  geom_line(group = 1) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  ylab("number of marriages")
```



Choice of model

Since the distribution is nowhere near linear we'll fit a model using polynomial regression. We want to avoid overfitting so we choose a degree of 3. Higher degrees will fit the data even better, but will scale less to other data.

```r
poly_model <- lm(nr_marriages ~ poly(age, 3), data = marriage_train)
```

Now let's make a data grid so we now where our data lies. In case of multiple variables this creates a combination of all the unique values for each variable. However, since we only use one independent variable age the unique combinations are just the ages.

```r
grid <- marriage_train %>%
  data_grid(age)
```

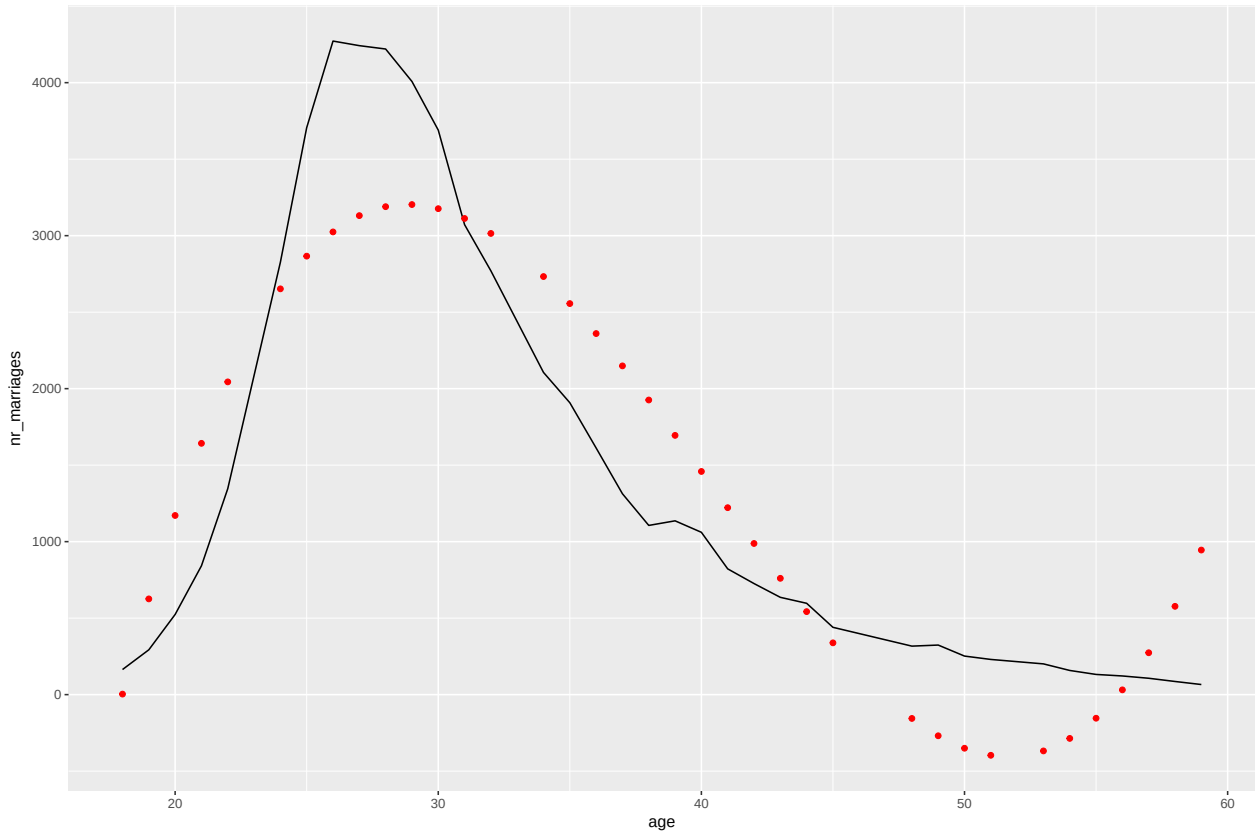
We add the predictions to the data grid:

```r
grid <- grid %>%
  add_predictions(poly_model)
```

```
marriage_train <- marriage_train %>%
  add_predictions(poly_model)
```

And let's see how well our model performed based on the predictions:

```
ggplot(data = marriage_train) +
  geom_line(aes(x = age, y = nr_marriages), group = 1) +
  geom_point(aes(x = age, y = pred), color = "red")
```



We can clearly see our model missed a rather large part of the pattern.