Forecast accuracy measures

EC 361–001

Prof. Santetti
Spring 2024

Materials

Required readings:

- `Hyndman & Athanasopoulos, ch. 6`
  - section 6.8.
  - sections 6.8 & 6.9 are *optional readings*.

Motivation

Motivation

One of the main challenges when estimating forecasting models is that we are trying to predict **unknown** (future) values of a variable.

We usually do not have the **luxury** of *waiting* until we see these future values so we can compare our predictions with the real data.

Fortunately, there are some techniques we can apply so we can evaluate the **accuracy** of our forecasts.

Point forecast accuracy

Point forecast accuracy

The *basic idea* of **forecast accuracy** measures is that we can evaluate how well a model performs on **new data** that were **not used** when fitting the model.

Given that, the procedure is to **split** the data set into two portions:

- The **training** set;
- The **test** set.

The **training** set is used to *estimate* any parameters of a forecasting method, while the **test** set is *left out* of the estimation step, being used to evaluate its *accuracy*.

The test set, then, should provide a **reliable indication** of how well the model is likely to forecast on new data.

Point forecast accuracy

In terms of the **size of the split**, a usual procedure is to leave **20%** of the total sample length for the test set.

Ideally, the test set should be *as large as the maximum forecast horizon required*.

Forecast errors

Forecast errors

A **forecast error** is the difference between an *observed* value and its *forecast*.

Formally,

$$e_{T+h} = y_{T+h} - \hat{y}_{T+h \mid T}$$

One **crucial** point:

- **Forecast errors are different from model residuals**!

Residuals are calculated on the *training* set while forecast **errors** are calculated on the *test* set.

Forecast errors

The first category of forecast error measures are the so-called **scale-dependent errors**.

These are located in the same **scale** as the original data (e.g., dollars, persons, percent).

The two main scale-dependent error measures are the **Mean Absolute Error (MAE)** and the **Root Mean Squared Error (RMSE)**.

$$\text{MAE} = \text{mean}(|e_t|)$$

$$\text{RMSE} = \sqrt{\text{mean}(e_t^2)}$$

When *comparing* different models, we would like to choose the one that **minimizes** theses errors.

Forecast errors

Another category of forecast errors are **percentage errors**.

These are **unit-free**, and thus can be compared across different data sets.

The most common percentage error measure is the **Mean Absolute Percentage Error (MAPE)**:

$$\text{MAPE} = \text{mean}(|p_t|)$$

where $p_t = 100e_t/y_t$.

One key **drawback** of this measure is that it is undefined for data points equal to zero $y_t = 0$.

Forecast errors

The last category involves **scaled errors**.

The main purpose of scaled errors is to provide comparison across different data sets (as with percentage errors), but not having the same issues as the previous two categories.

For a **non-seasonal** time series, a useful way to define a scaled error uses *naïve forecasts*:

$$q_j = \frac{e_j}{\frac{1}{T-1}\sum_{t=2}^{T} |y_t - y_{t-1}|}$$

For **seasonal data**, the formula looks like the following (*m* is the seasonal period):

$$q_j = \frac{e_j}{\frac{1}{T-m}\sum_{t=m+1}^{T} |y_t - y_{t-m}|}$$

Forecast errors

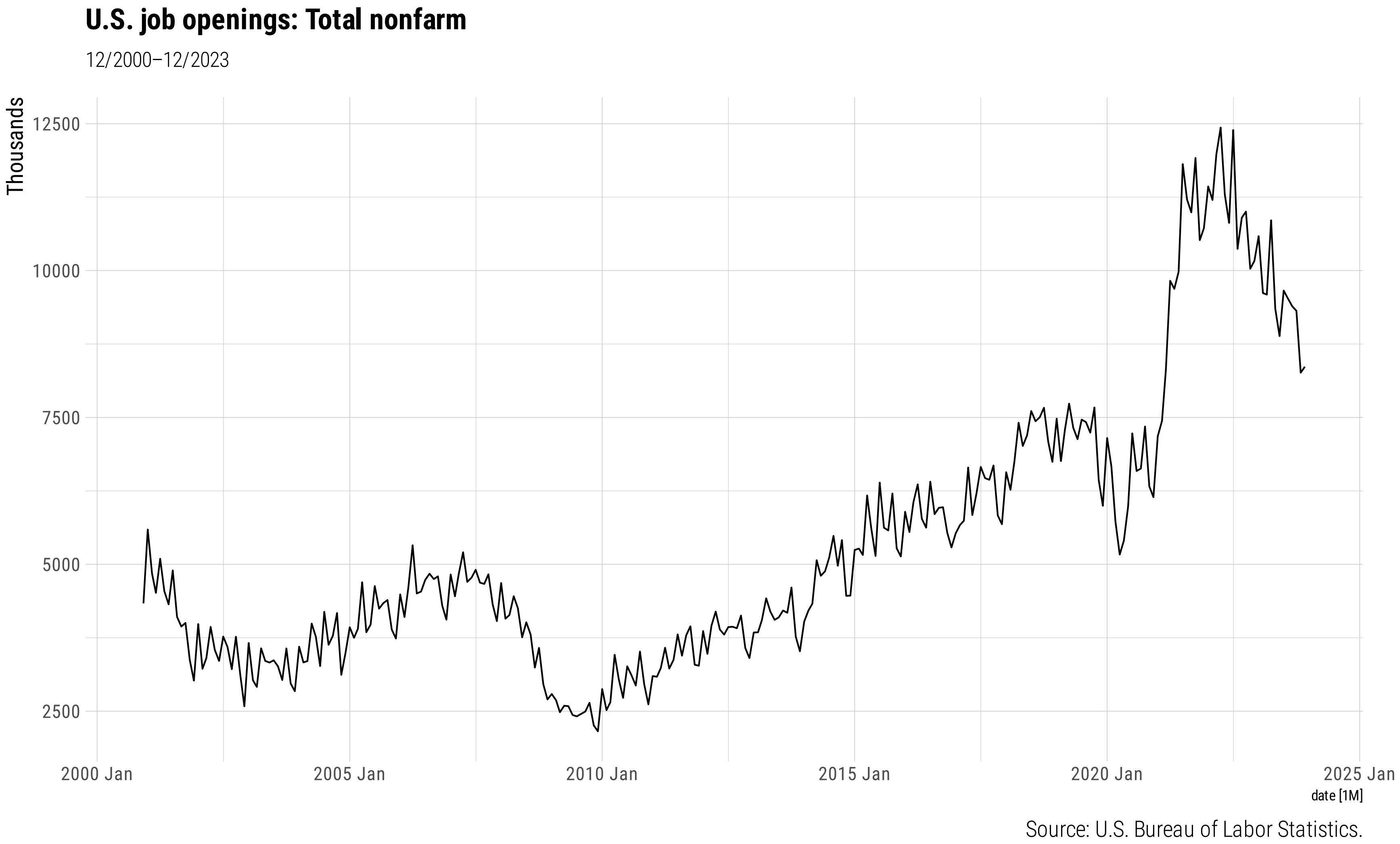
The value of the error q_j computed by the previous formula is **scale-free**.

Then, we are able to calculate the **Mean Absolute Scaled Error (MASE)**:

$$\mathrm{MASE} = \mathrm{mean}(|q_j|)$$

An example

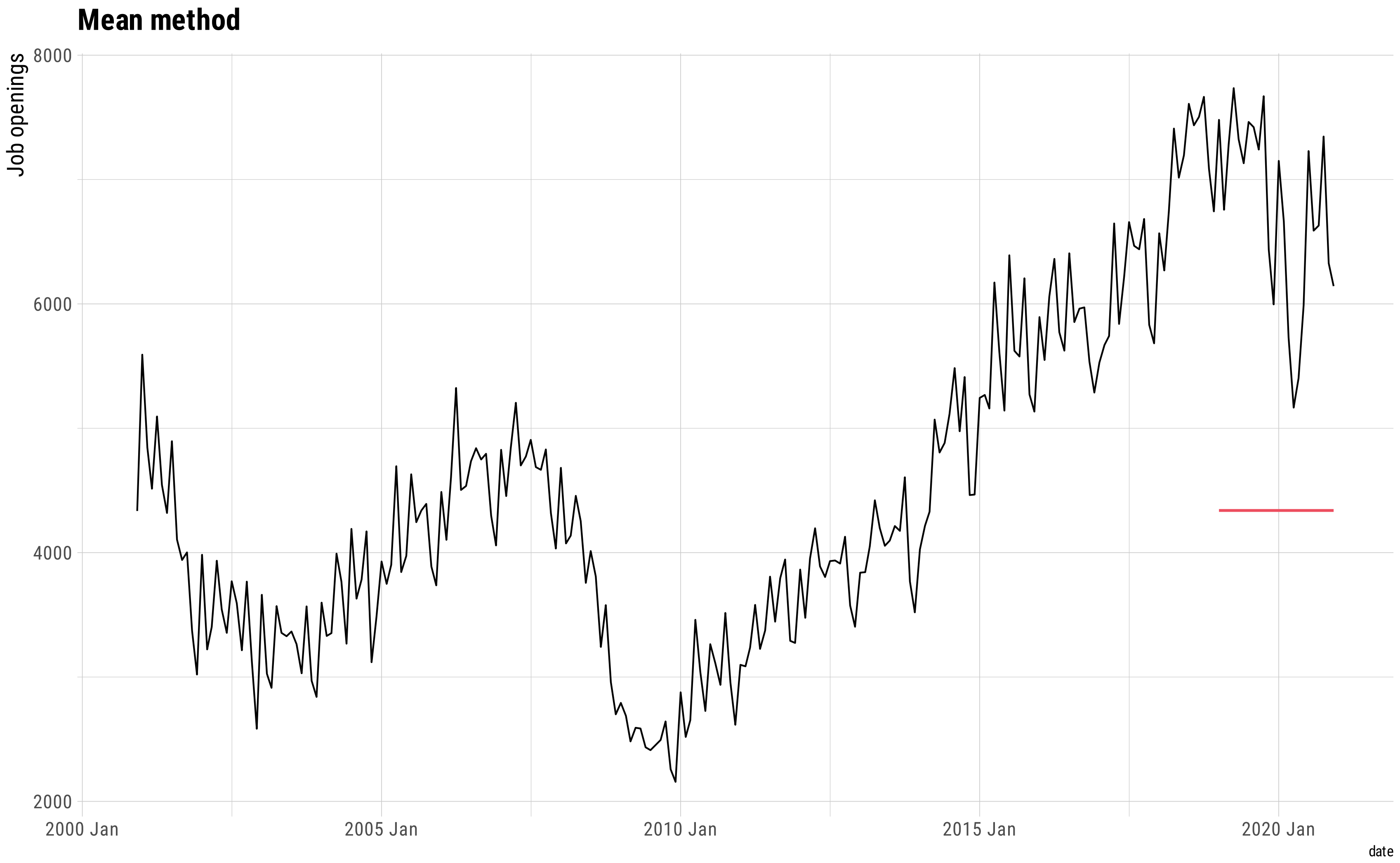
An example


U.S. job openings: Total nonfarm
12/2000–12/2023
Thousands
12500
10000
7500
5000
2500
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
2025 Jan
date [1M]
Source: U.S. Bureau of Labor Statistics.

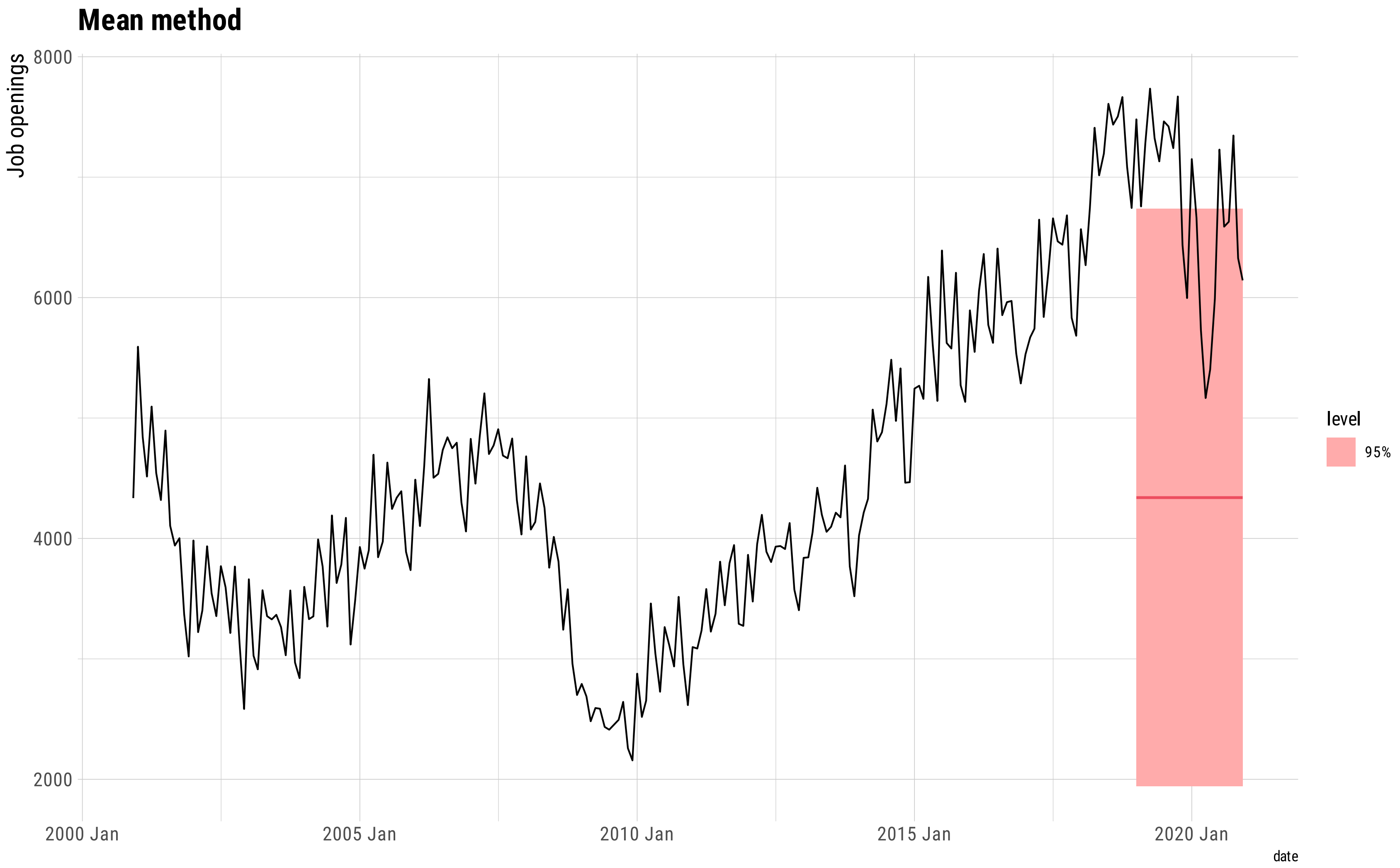
An example

```r
## Defining the training set:

job_train <- job_ts |>
  filter_index(. ~ "2018-12-01")

## Estimating several benchmark models:

job_fit <- job_train |>
  model(mean_model = MEAN(openings),
        naive_model = NAIVE(openings),
        snaive_model = SNAIVE(openings),
        drift_model = RW(openings ~ drift()),
        snaive_with_drift = RW(openings ~ drift() + lag(12)))

## 24-month ahead forecast:

job_fc <- job_fit |>
  forecast(h = 24)
```

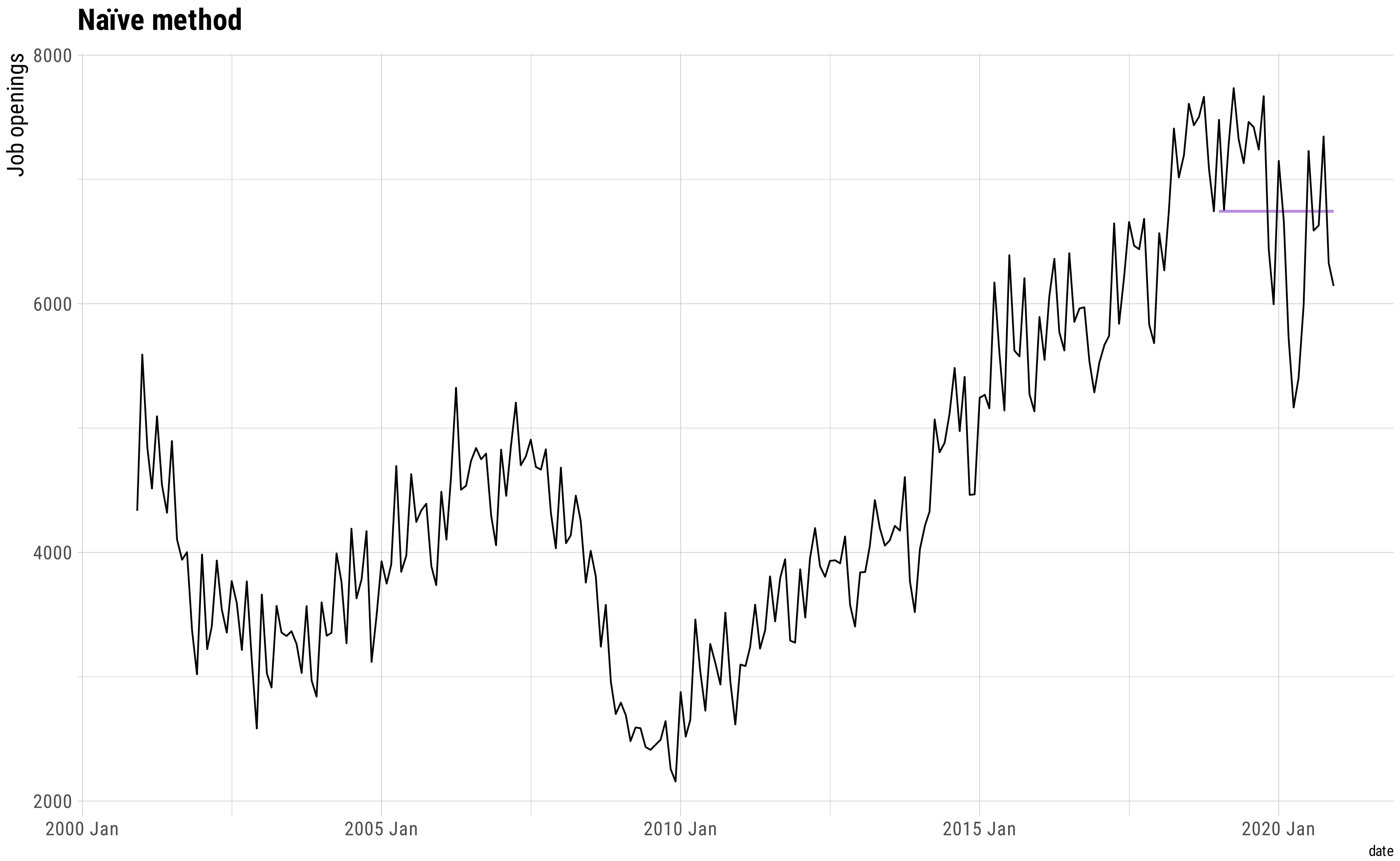
An example


Mean method
Job openings
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date

An example

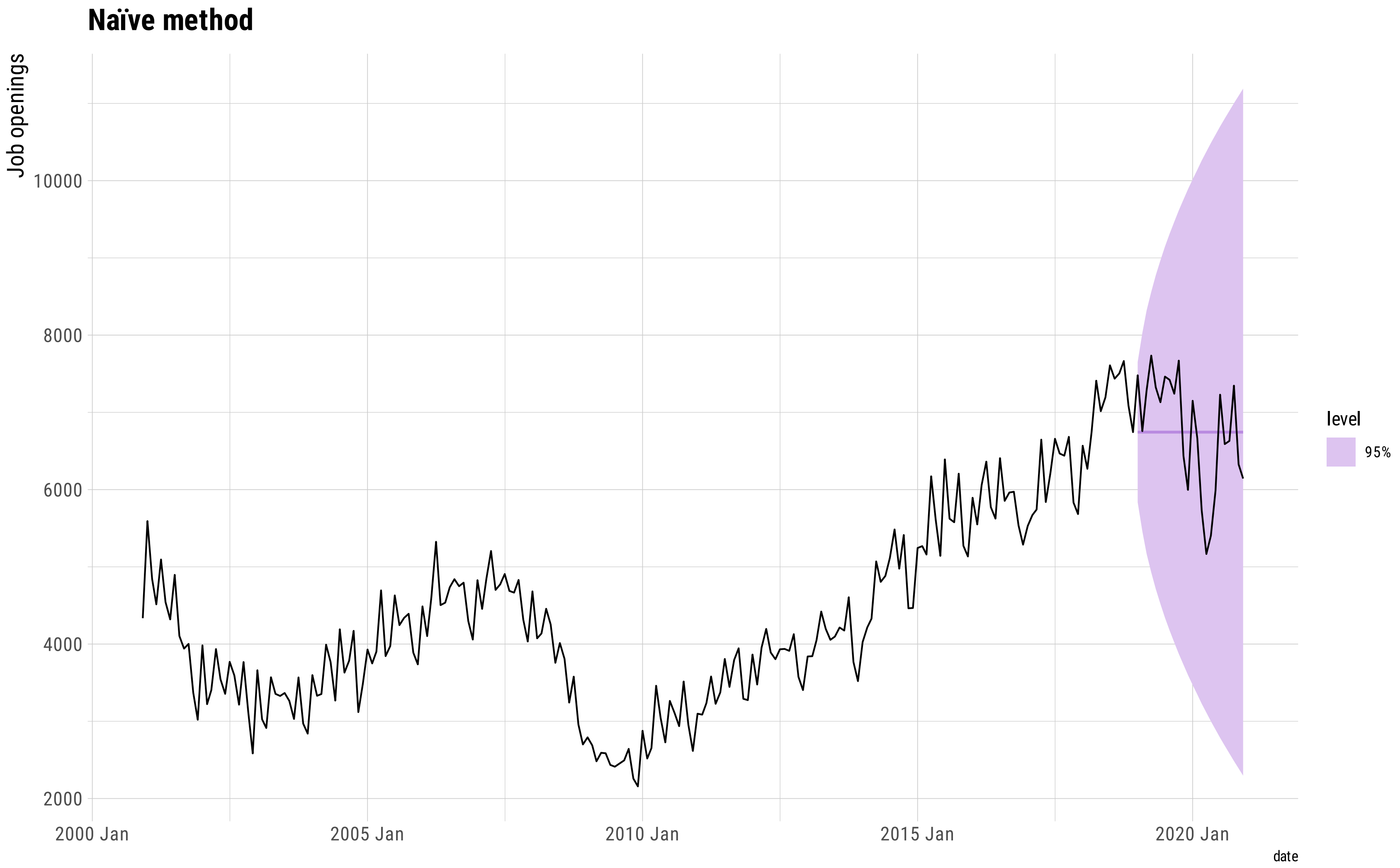

Mean method
Job openings
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date
level
95%

An example

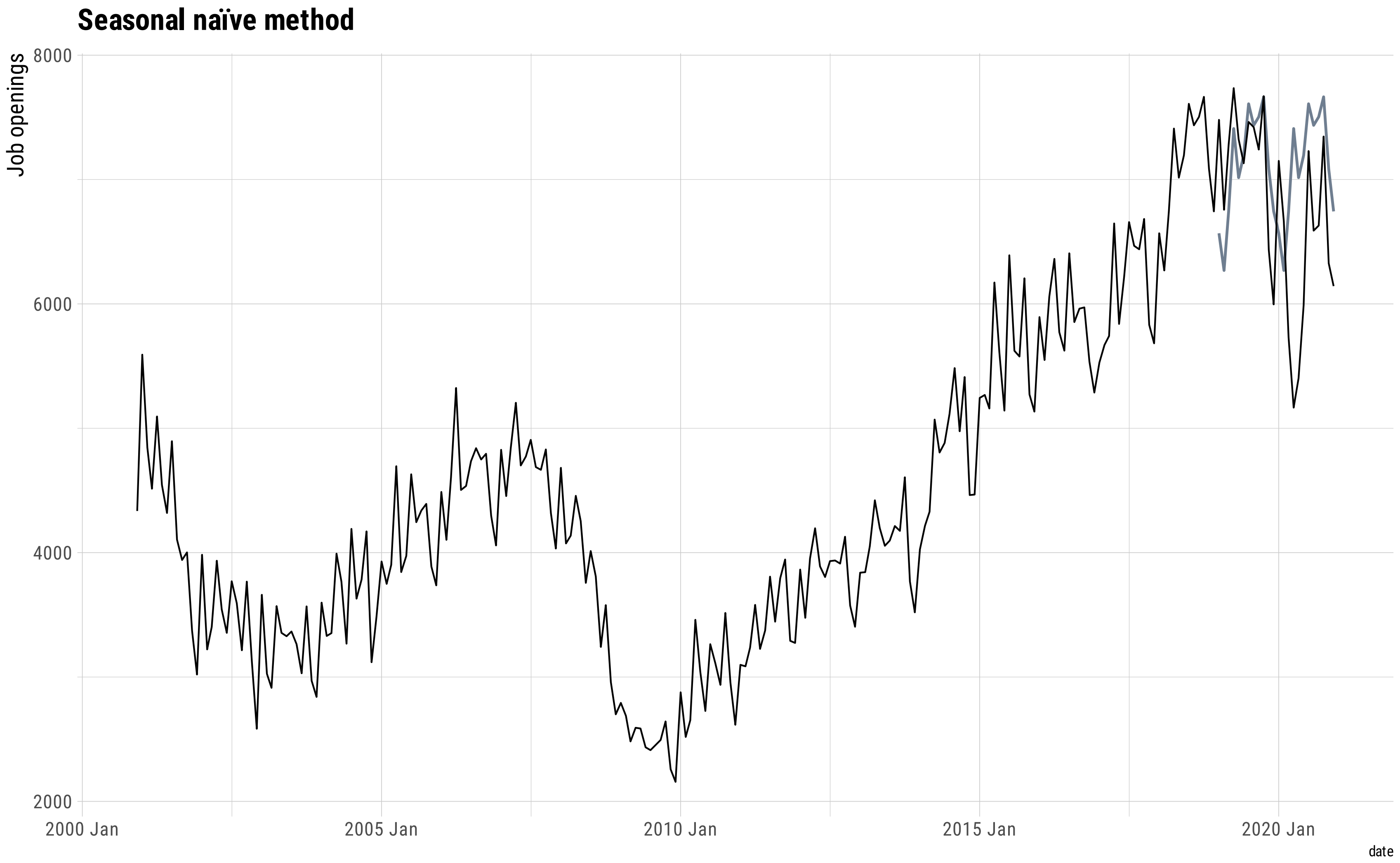


Naïve method
Job openings
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date

An example

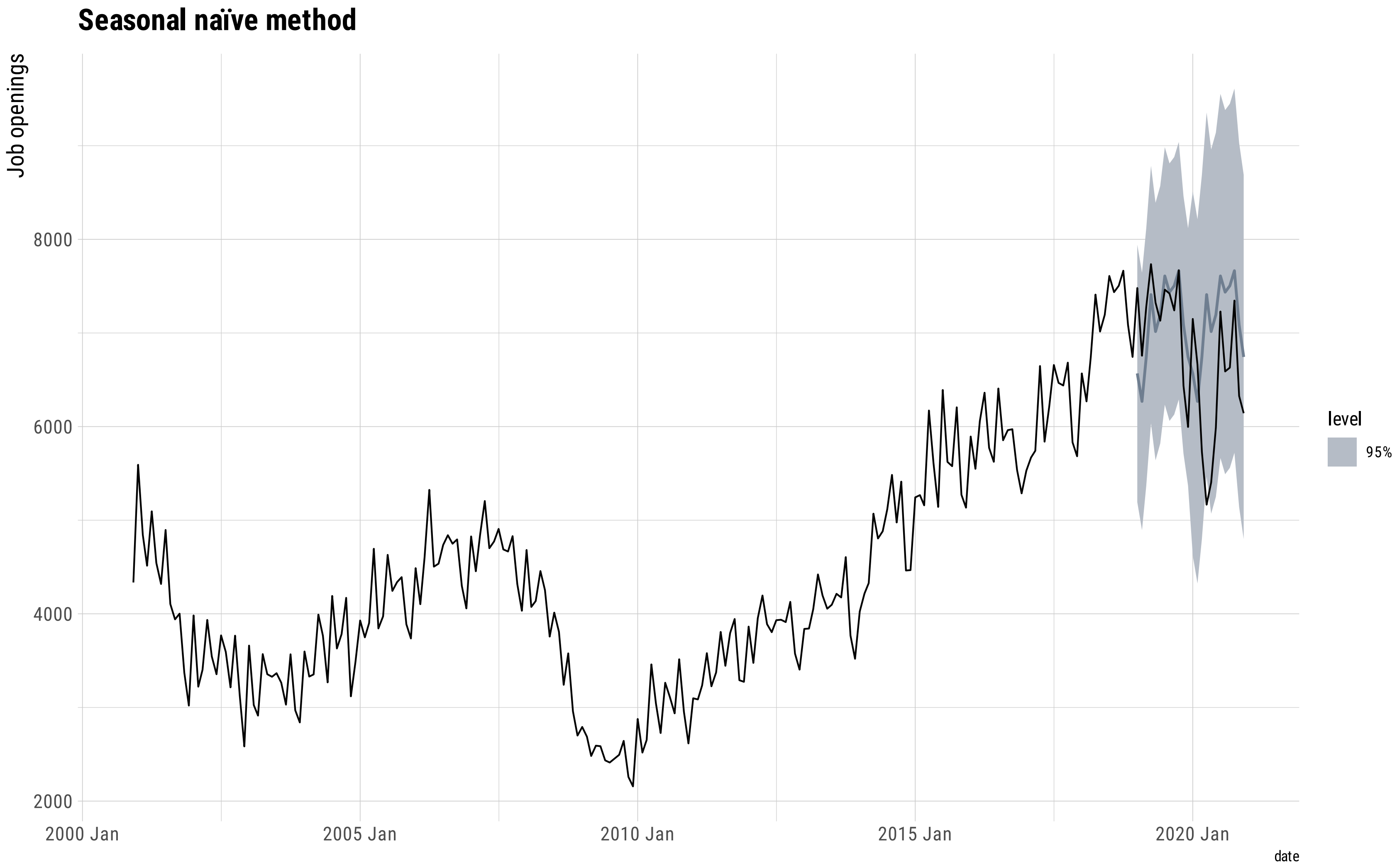

Naïve method
Job openings
10000
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date
level
95%

An example



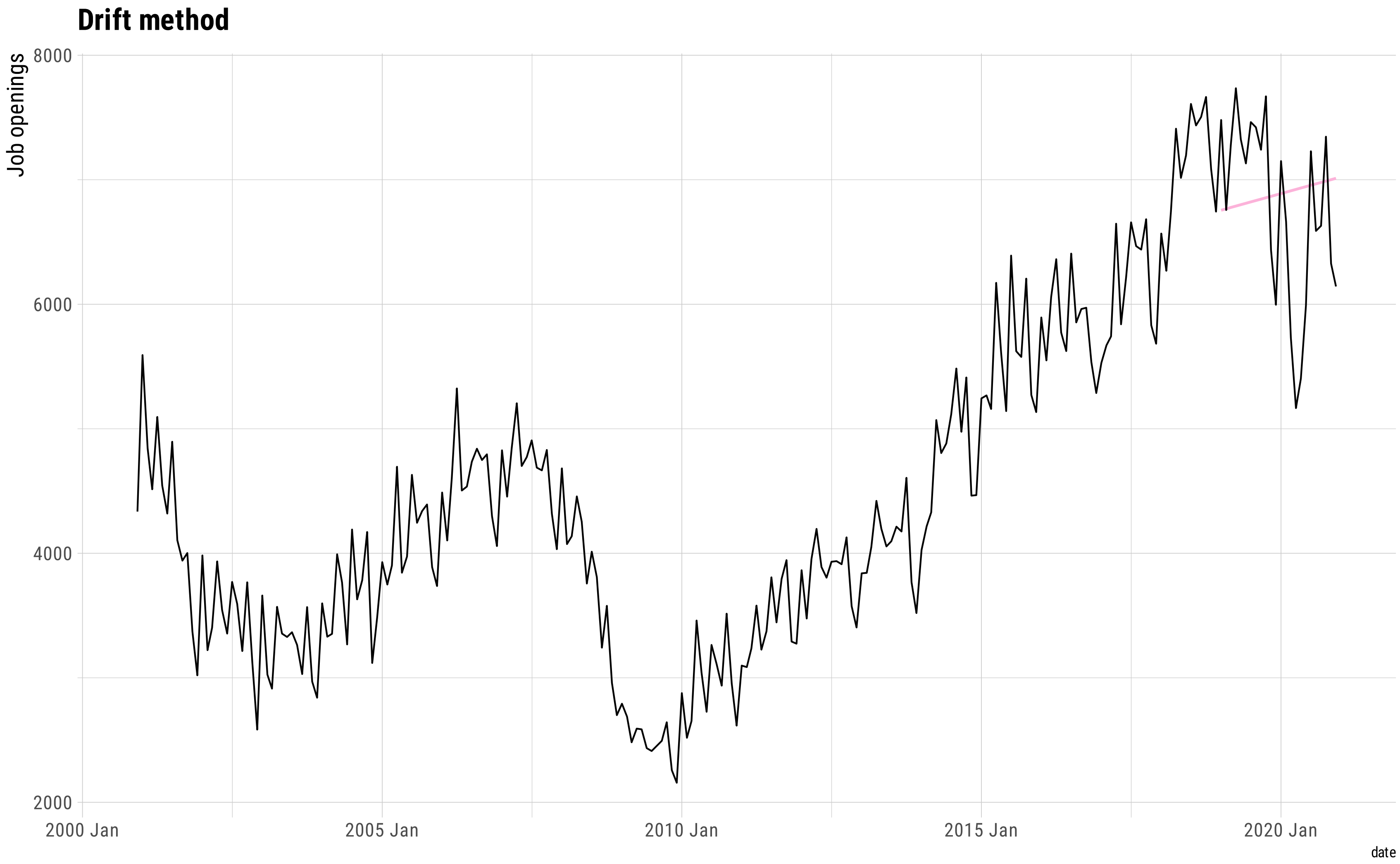
Seasonal naïve method
Job openings
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date

An example



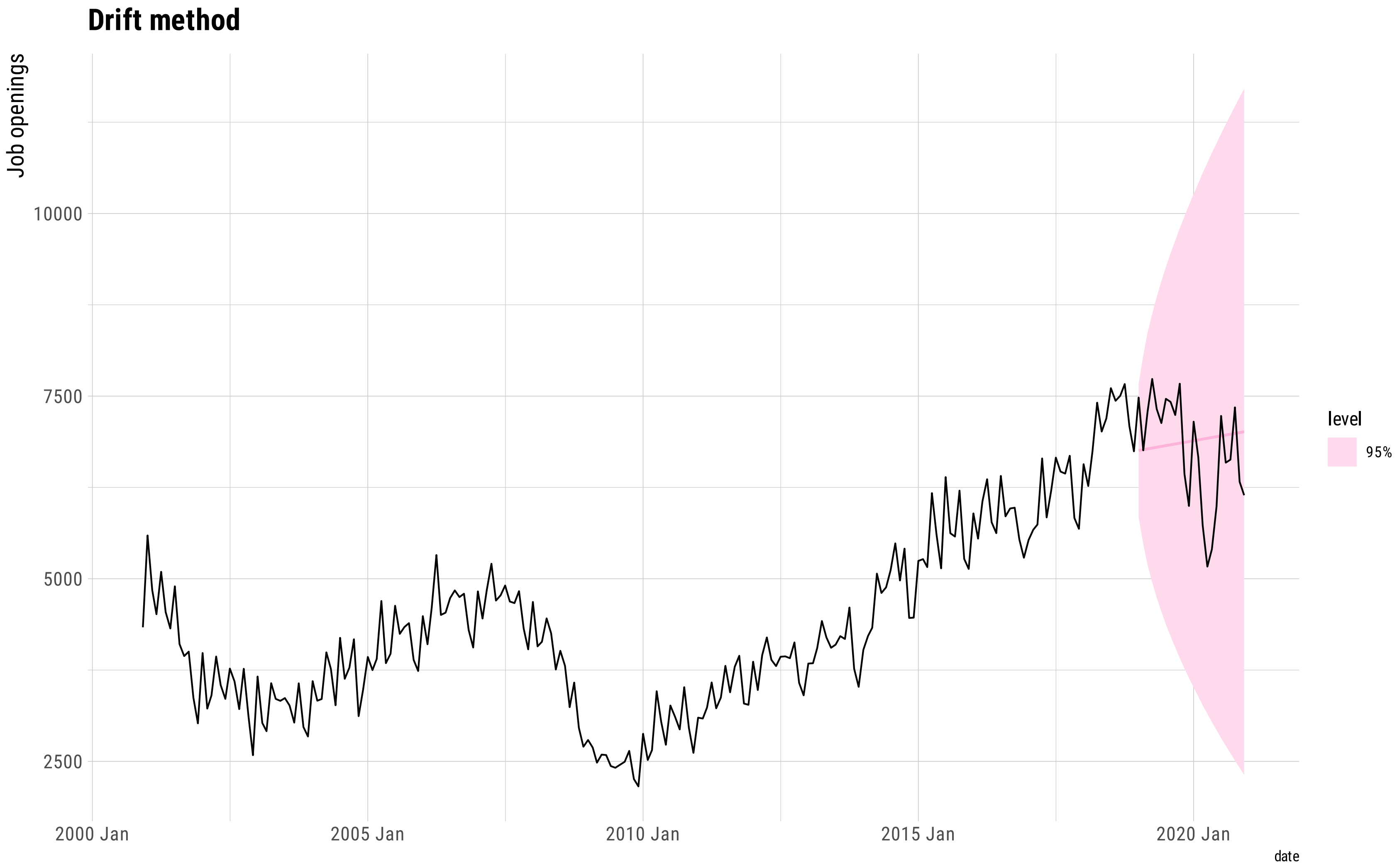
Seasonal naïve method
Job openings
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date
level
95%

An example



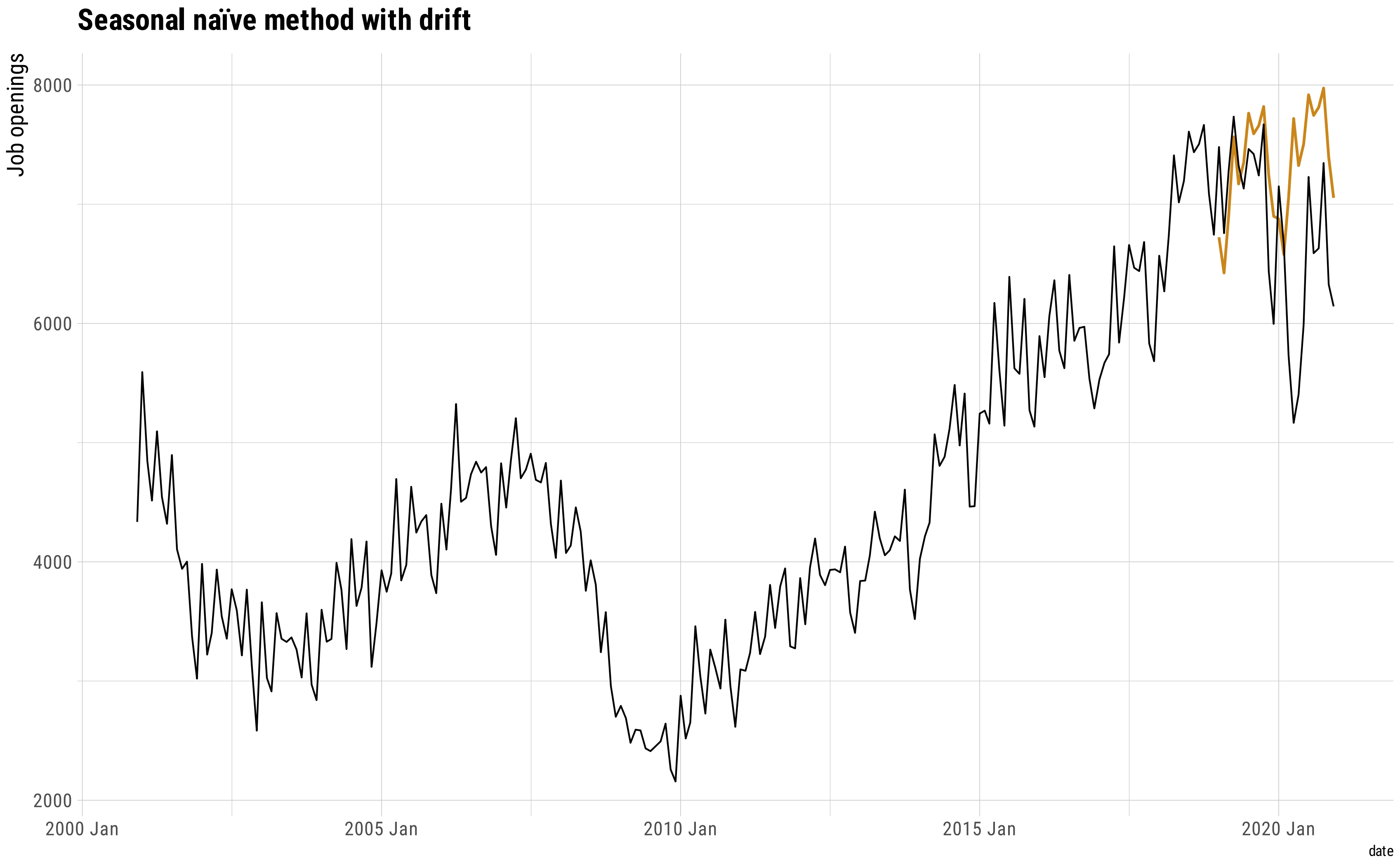
Drift method
Job openings
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date

An example



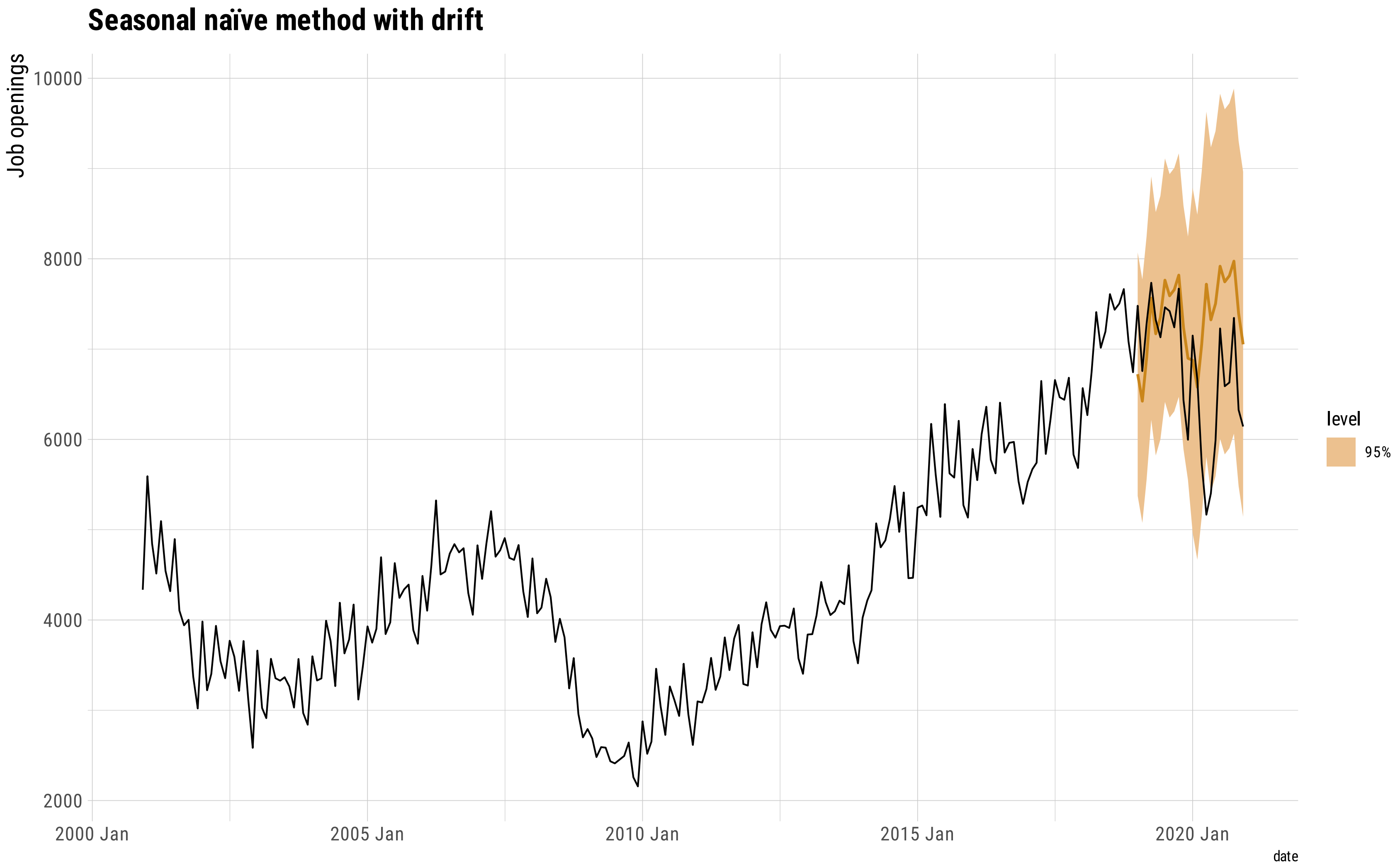
Drift method
Job openings
10000
7500
5000
2500
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date
level
95%

An example



Seasonal naïve method with drift
Job openings
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date

An example


Seasonal naïve method with drift
Job openings
10000
8000
6000
4000
2000
2000 Jan
2005 Jan
2010 Jan
2015 Jan
2020 Jan
date
level
95%

An example

```
## All accuracy measures:

job_fc ▷
  accuracy(job_ts)
```

```
#> # A tibble: 5 × 10
#>   .model            .type     ME  RMSE   MAE    MPE  MAPE  MASE RMSSE  ACF1
#>   <chr>             <chr>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1 drift_model       Test  -121.   767.  650. -3.10  10.2   1.14  1.09 0.592
#> 2 mean_model        Test  2423.  2527. 2423. 35.1   35.1   4.24  3.60 0.557
#> 3 naive_model       Test    18.5  715.  611. -0.947  9.43  1.07  1.02 0.557
#> 4 snaive_model      Test  -341.   814.  637. -6.18  10.3   1.11  1.16 0.623
#> 5 snaive_with_drift Test  -573.   971.  753. -9.73  12.2   1.32  1.38 0.669
```

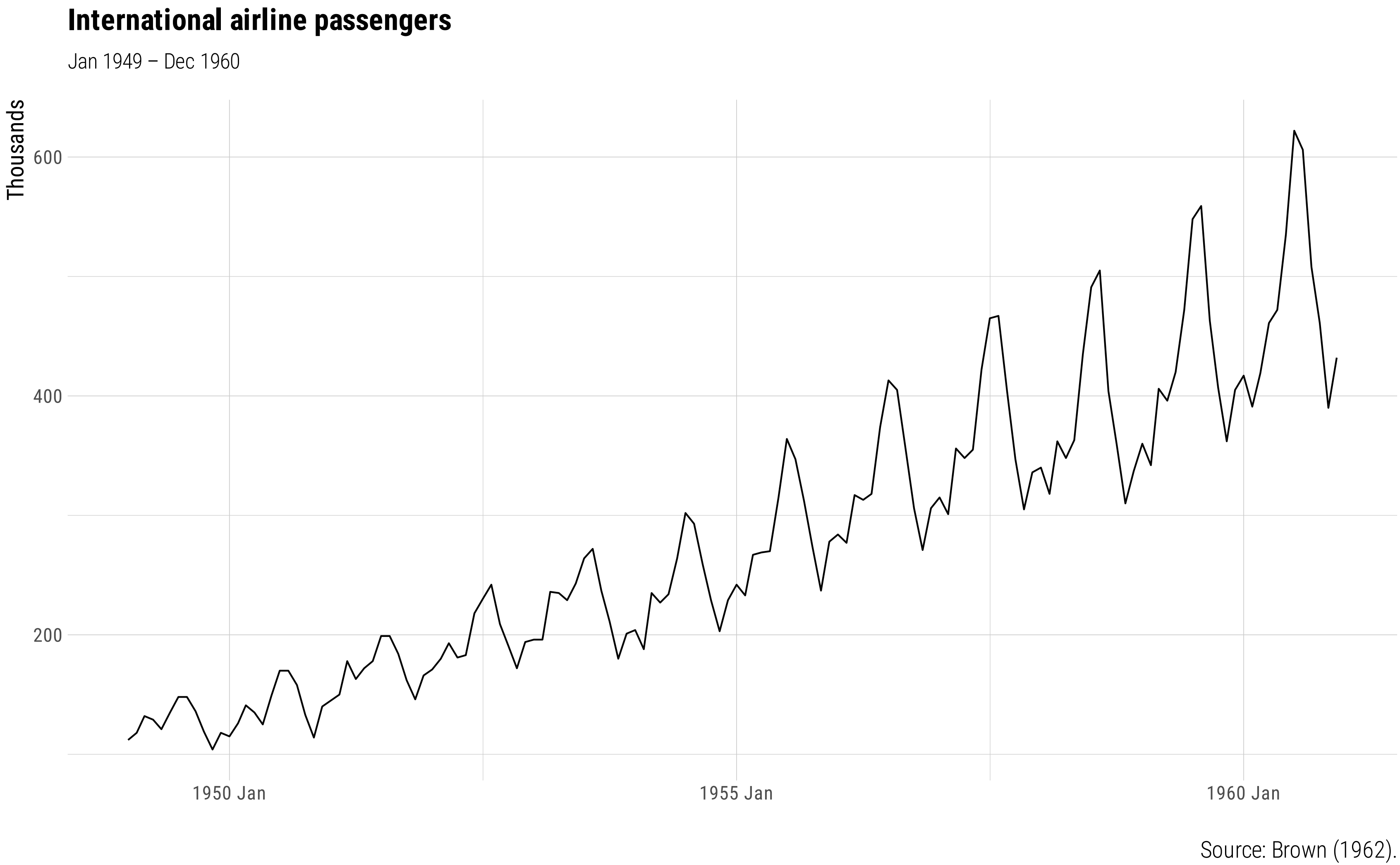
An example

```
## Main accuracy measures:

job_fc ▷
  accuracy(job_ts) ▷
  select(.model, MAE, RMSE, MAPE, MASE)
```

```
#> # A tibble: 5 × 5
#>   .model               MAE  RMSE  MAPE  MASE
#>   <chr>              <dbl> <dbl> <dbl> <dbl>
#> 1 drift_model         650.  767. 10.2   1.14
#> 2 mean_model         2423. 2527. 35.1   4.24
#> 3 naive_model         611.  715.  9.43  1.07
#> 4 snaive_model        637.  814. 10.3   1.11
#> 5 snaive_with_drift   753.  971. 12.2   1.32
```

Another example

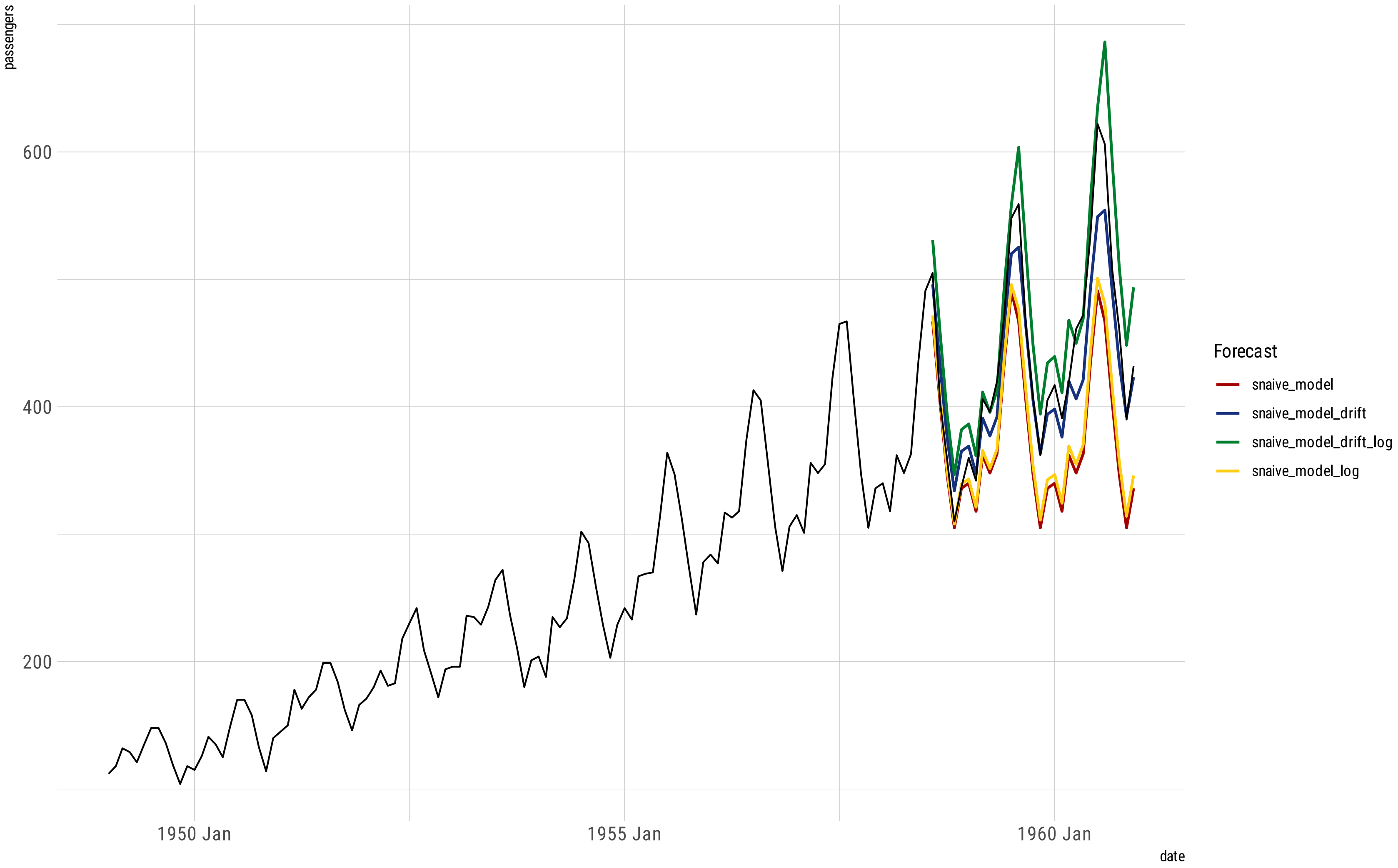
Another example



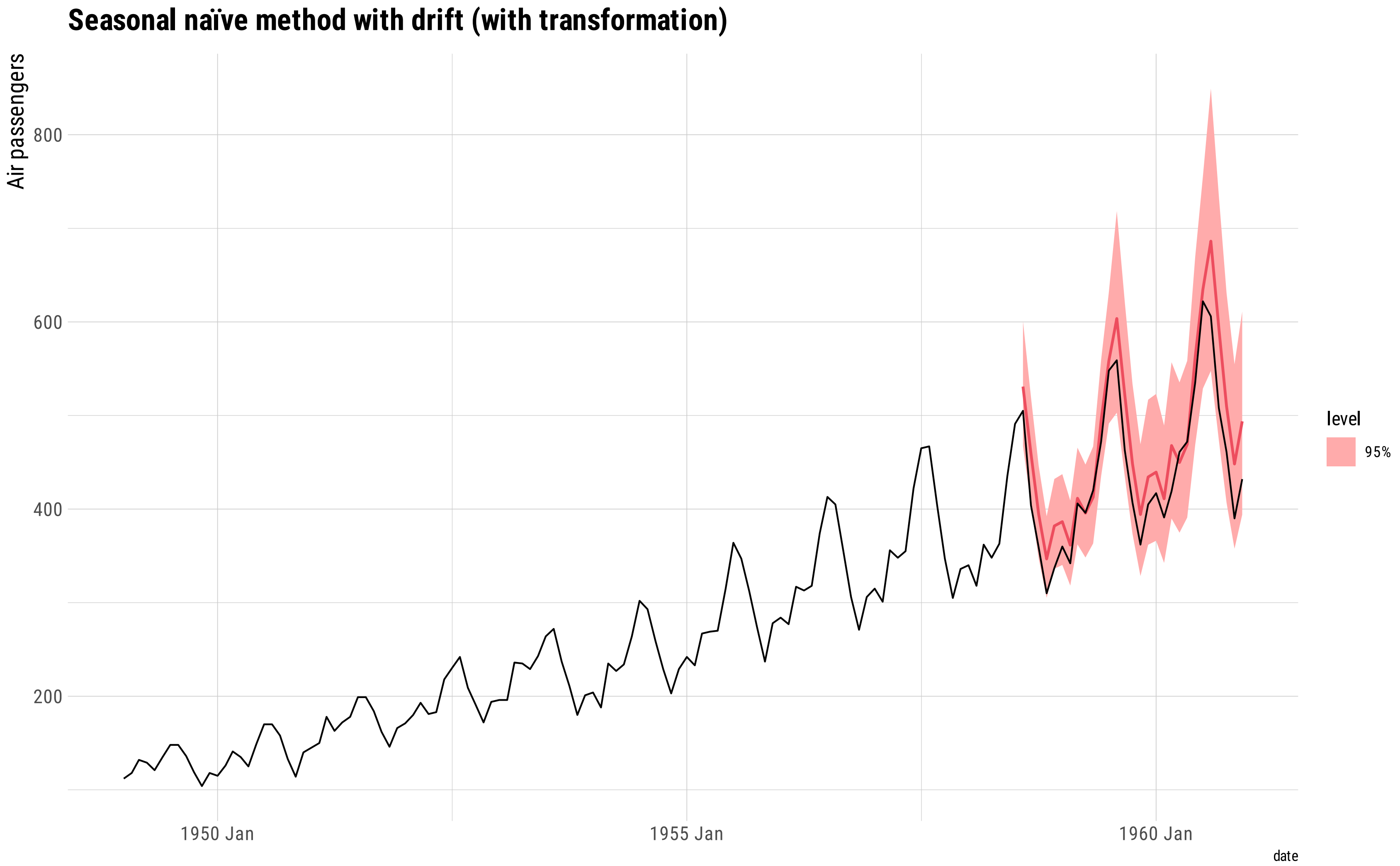
Another example

```
## Training set:

air_train <- air_ts |>
  filter_index(. ~ "1958-07-01")

## Fitting different models:

air_fit <- air_train |>
  model(snaive_model_drift_log = RW(log(passengers) ~ drift() + lag(12)),
        snaive_model_log = SNAIVE(log(passengers)),
        snaive_model_drift = RW(passengers ~ drift() + lag(12)),
        snaive_model = SNAIVE(passengers))

## Forecasting 29 months ahead:

air_fc <- air_fit |>
  forecast(h = 29)
```

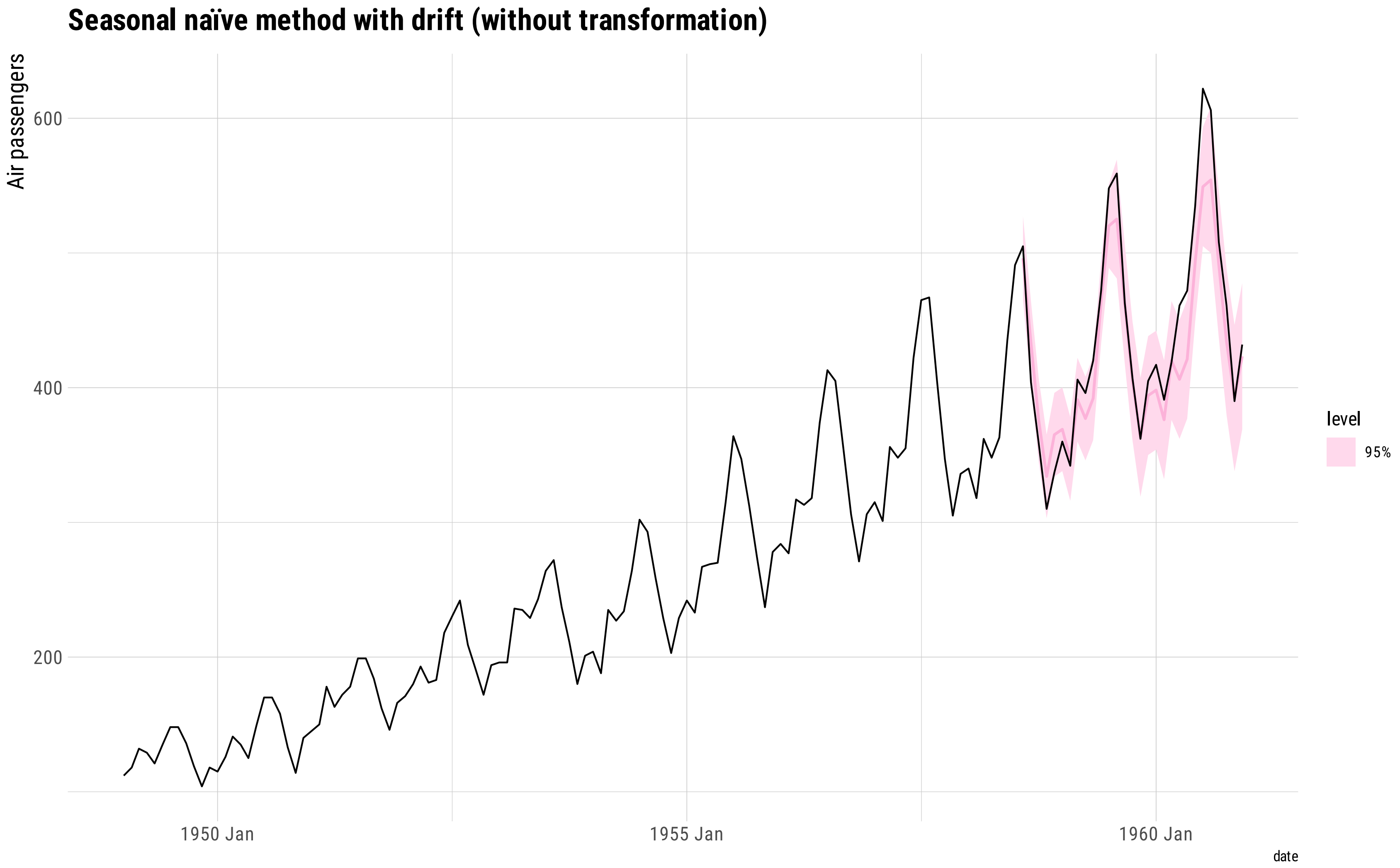
Another example



Another example



Another example



Another example

```
air_fc ▷
  accuracy(air_ts) ▷
  select(.model, MAE, RMSE, MAPE, MASE)
```

```
#> # A tibble: 4 × 5
#>   .model                   MAE  RMSE  MAPE  MASE
#>   <chr>                  <dbl> <dbl> <dbl> <dbl>
#> 1 snaive_model            64.8  75.2 14.0  2.20
#> 2 snaive_model_drift      21.7  28.2  4.70 0.737
#> 3 snaive_model_drift_log  33.7  40.2  7.80 1.15
#> 4 snaive_model_log        58.6  68.3 12.7  1.99
```

Next time: Exponential smoothing