Group Assignment 4: Prediction Competition

Paige Steffan, Spencer Wood, Sydney Johnston, Rose Civitts

Data Preparation

- To prepare the data we split it into the holdout and training groups
  - credit_ts <- credit_ts %>%filter(year(month) >= '1970 Feb')
  - TrainingCredit <- credit_ts %>% filter(year(month) <= '2004 Jan')
- In our graph you can see the first snippet of the training group we have used.
  - We made the training group to be anything less than or in January 2004 and the hold out to be anything earlier than that.
  - Splitting the data allows us to have accurate models that are relevant to the data we collect in the future.
- After creating these groups, we were able to analyze our data.

credit_in_millions	month
1.0135	1971 Jan
1.2312	1971 Feb
1.1437	1971 Mar
1.2207	1971 Apr
1.1890	1971 May
1.1804	1971 Jun
1.1453	1971 Jul
1.1166	1971 Aug
1.0440	1971 Sep
1.1281	1971 Oct
0.9875	1971 Nov
0.9661	1971 Dec
1.0216	1972 Jan
1.0930	1972 Feb
0.9890	1972 Mar
0.8717	1972 Apr
1.0888	1972 May
0.9883	1972 Jun
1.0371	1972 Jul
1.0265	1972 Aug
1.0217	1972 Sep
1.0685	1972 Oct
0.9290	1972 Nov

Adding Time Series Component

In order for the credits data to be valid for time series analysis, a time component must be added.

We know the credit data was collected on a monthly basis and are making the assumption that data was collected each month (no skipped months) and the months were sequential.

```
credit$month <- 492:1
credit$month <- yearmonth(credit$month)
```

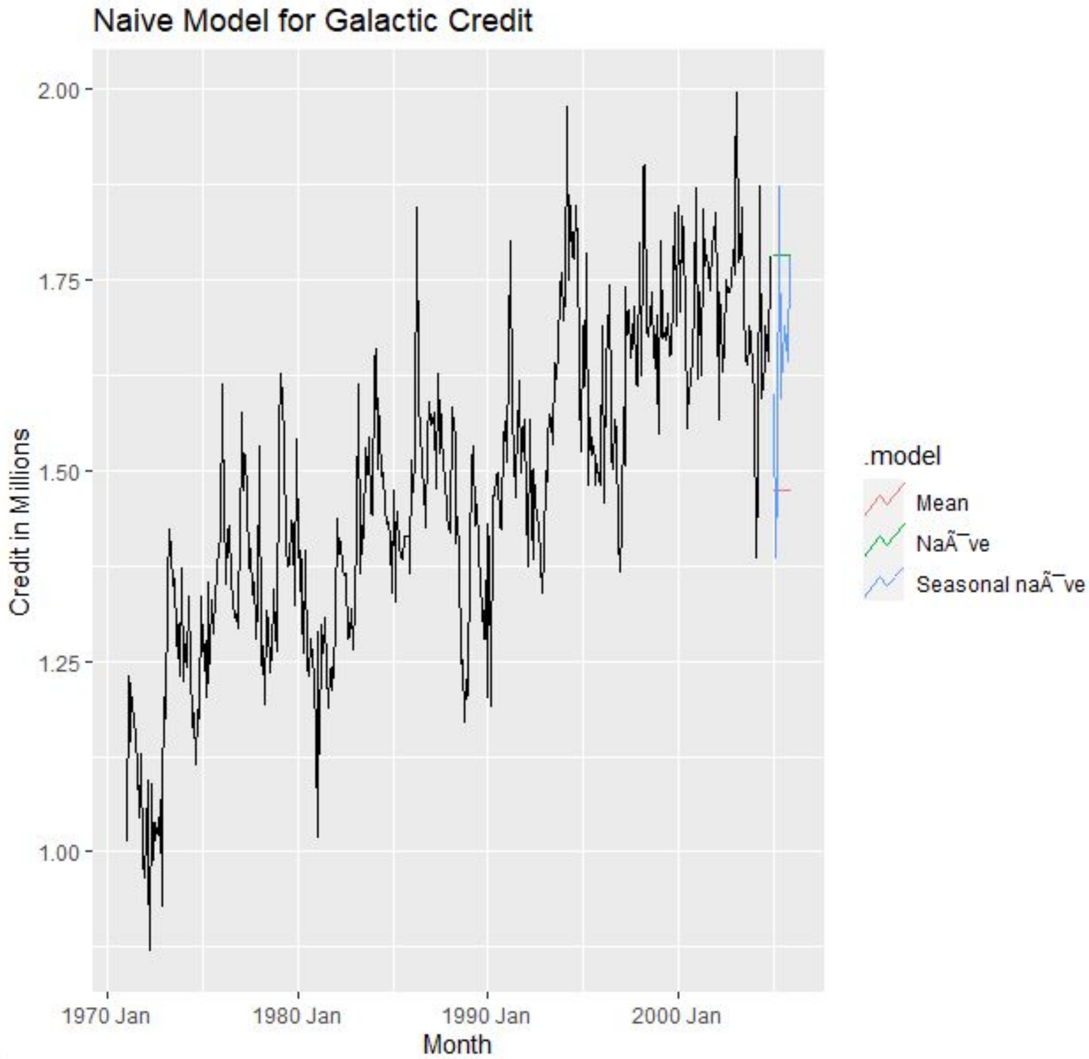
This code was used to assign sequential months to the data. Since the data is sequential by month, the year chosen is not important in forecasting or analysis.

Naive Model

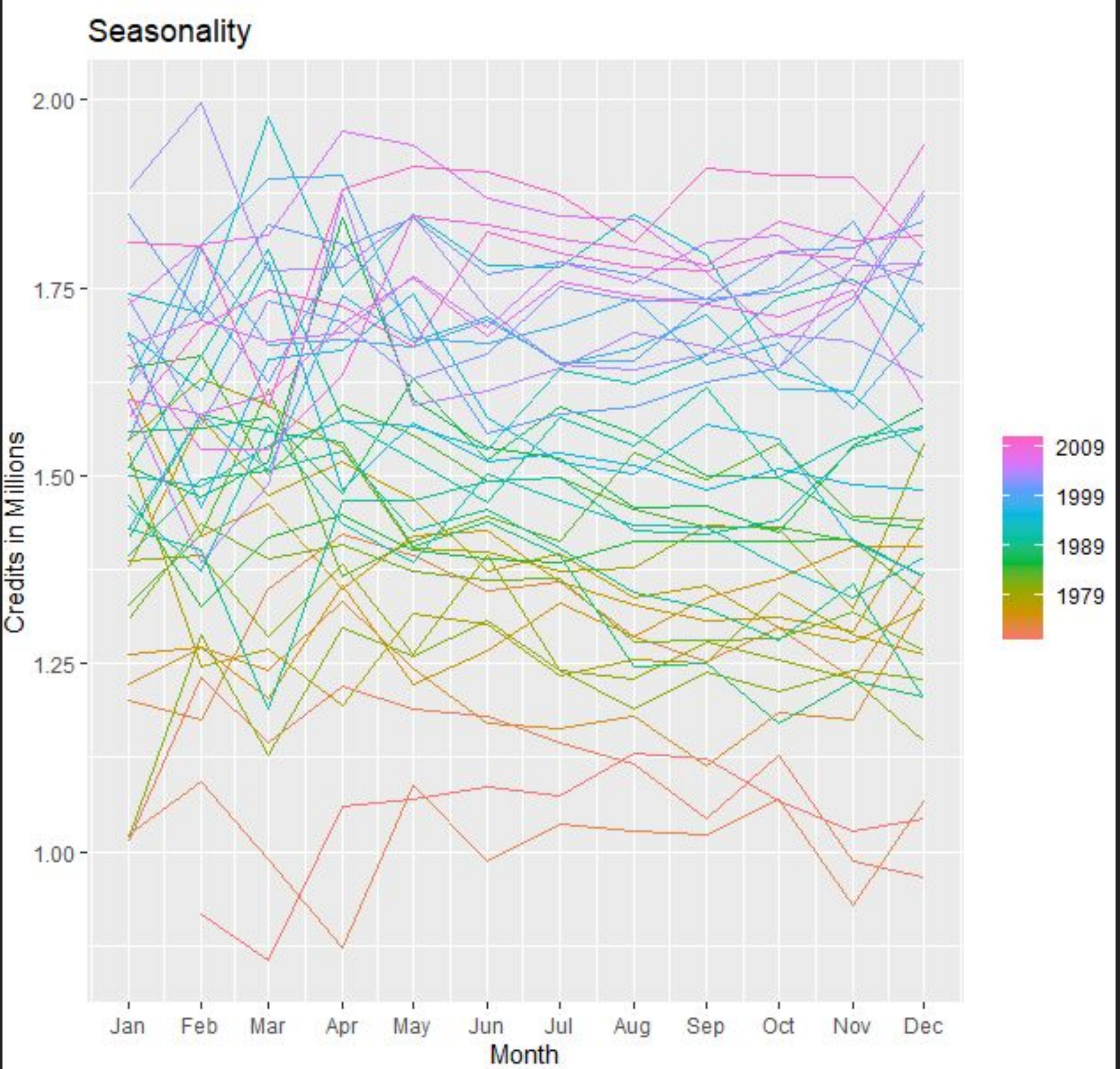
- Forecasting where the last periods sales are used for the next forecast without predictions or adjusting factors
    - Uses current value or mean for next forecast
    - For the most part, this economically works very well.

Naive Model Interpretation

- In our case, it looks as if only the seasonal naive credits will be close to last months.
- Since the mean method and the naive method do not follow the trend, we can say that the seasonal naive model is best here.



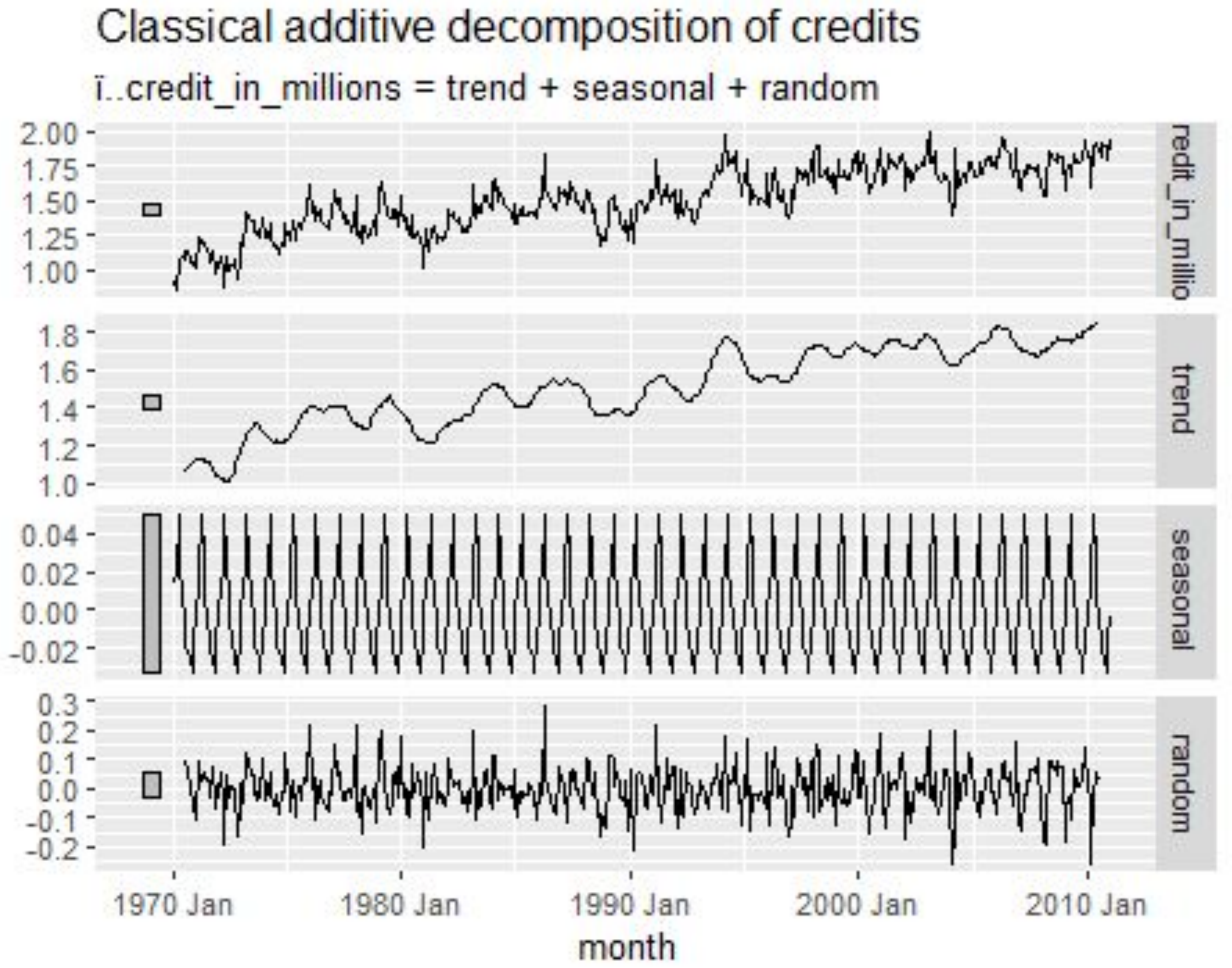
Credits Seasonal Plot



Seasonal Plot Interpretation

- It is hard to pick out the seasonality in the data from this graph, but one thing to notice is that most years seem to drop low in February and March, rise rapidly in April and May, then slowly decline through the rest of the year.
- The randomness of the data stands out in this graph, and is also important in our next graph, the decomposition of the data.

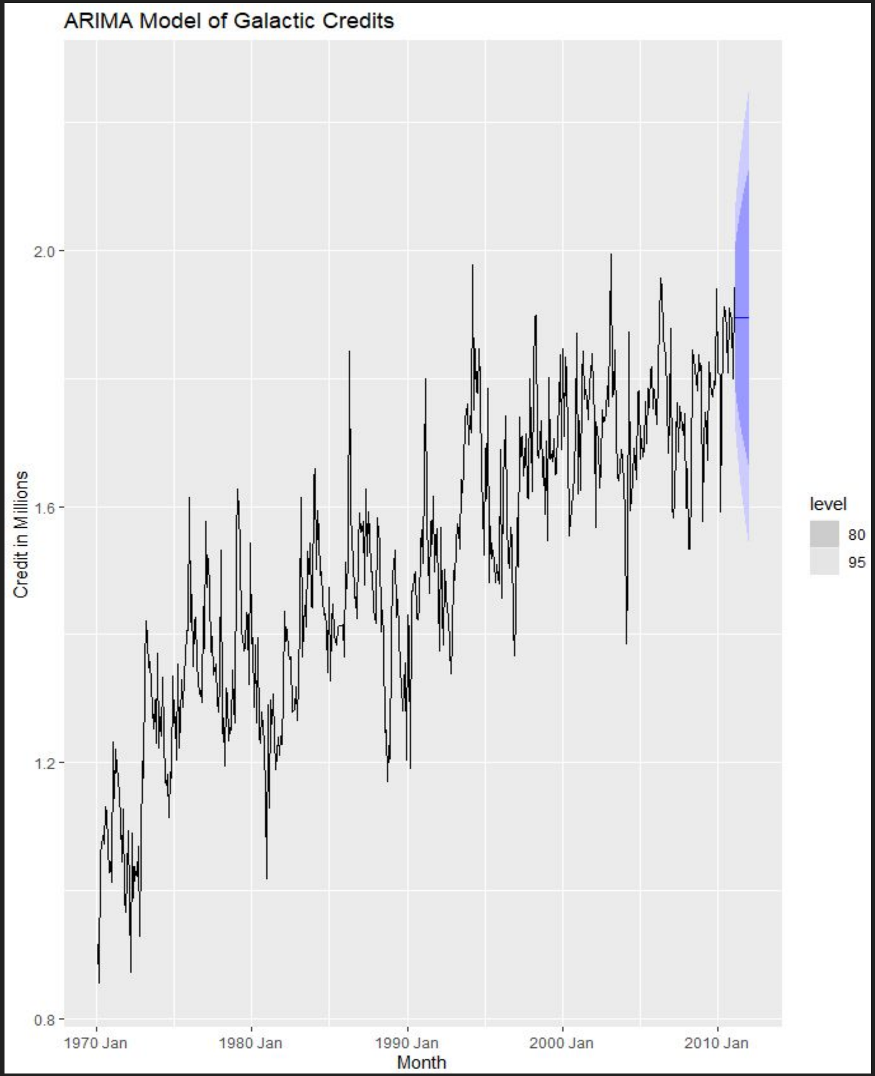
Classical Additive Decomposition



Decomposition Model Breakdown

- The decomposition of the data set shows that the trend has the largest impact on the data, shown by it having the smallest bar on the scale to the far left of the graphs.
- Seasonality is shown to have a much weaker impact on the data because the scale on the left is drastically larger than the rest.
- The random “white noise” of the data has more of an impact on the data than the seasonality, which is not great. Ideally the data would not be as random so that our forecasting power would be greater.

ARIMA Model


ARIMA Model of Galactic Credits
Credit in Millions
2.0
1.6
1.2
0.8
1970 Jan
1980 Jan
1990 Jan
2000 Jan
2010 Jan
Month
level
80
95

ARIMA Model Interpretation

- ARIMA stands for “autoregressive integrated moving average”. In simpler terms, it uses past data to predict future data for a time series.

Five ARIMA models were used to find the best model for the series. The best fit is the stepwise ARIMA(0,1,1)

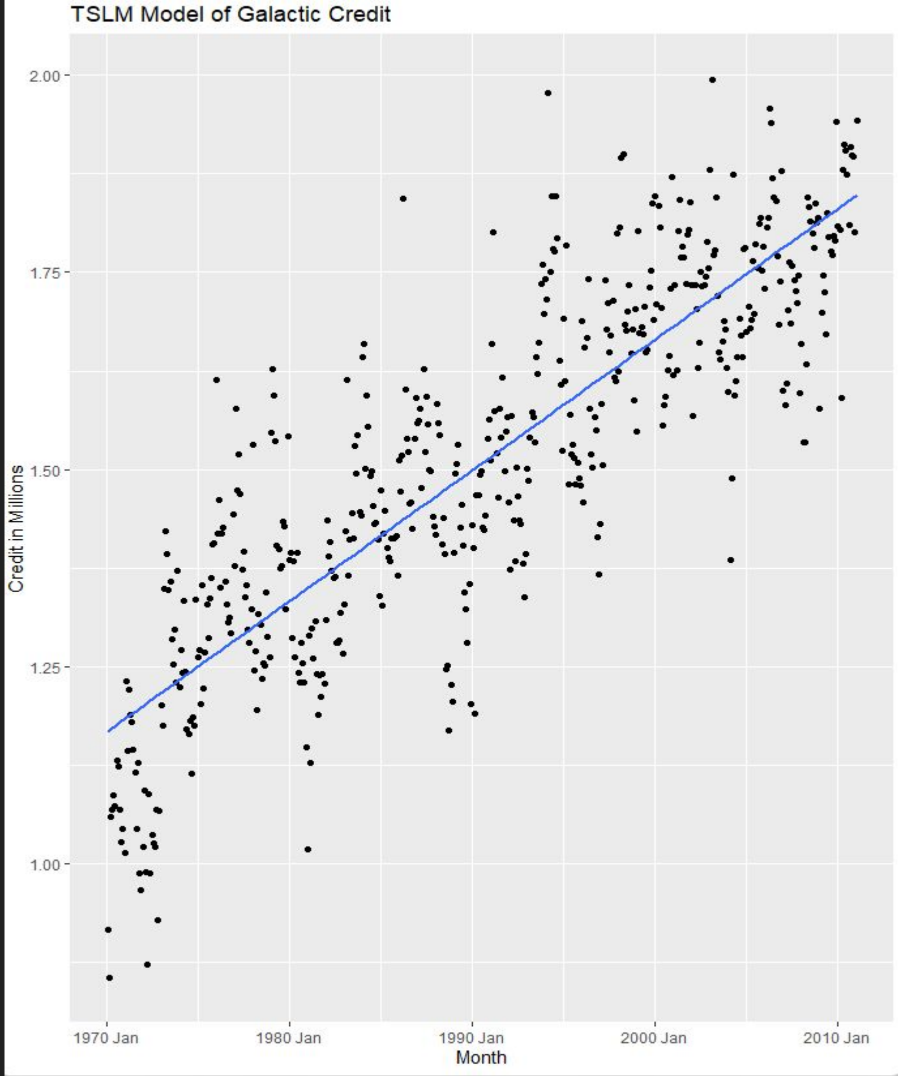
```
> glance(fit) %>% arrange(AICc)
# A tibble: 5 x 8
  .model           sigma2 log_lik   AIC  AICc   BIC ar_roots   ma_roots
  <chr>             <dbl>   <dbl> <dbl> <dbl> <dbl> <list>     <list>
1 searching       0.00789    494. -974. -974. -945. <cpl [14]> <cpl [25]>
2 arima2          0.00827    482. -956. -956. -939. <cpl [0]>  <cpl [3]>
3 arimastepwise   0.00835    479. -953. -953. -945. <cpl [0]>  <cpl [1]>
4 arima3          0.00833    481. -950. -950. -925. <cpl [2]>  <cpl [2]>
5 arima1          0.00849    475. -944. -944. -931. <cpl [2]>  <cpl [0]>
> fit <- credit_ts %>% model(ARIMA())
Model not specified, defaulting to automatic modelling of the `ï..credit_in_millions` variable.
rride this using the model formula.
> report(fit)
Series: ï..credit_in_millions
Model: ARIMA(0,1,1)
```

The AICc for the (0,1,1) model is -953.08 and the RMSE is .0912

```
> report(fit)
Series: ï..credit_in_millions
Model: ARIMA(0,1,1)

Coefficients:
          ma1
       -0.470
s.e.    0.044

sigma^2 estimated as 0.008349:  log likelihood=478.55
AIC=-953.1   AICc=-953.08   BIC=-944.71
```

```
  .model  .type          ME   RMSE    MAE     MPE  MAPE  MASE RMSSE    ACF1
  <chr>   <chr>       <dbl>  <dbl>  <dbl>   <dbl> <dbl> <dbl> <dbl>   <dbl>
1 ARIMA() Training 0.00377 0.0912 0.0691 0.00142  4.65 0.530 0.552 0.00845
```

TSLM Model


TSLM Model of Galactic Credit
2.00
1.75
1.50
1.25
1.00
Credit in Millions
1970 Jan
1980 Jan
1990 Jan
2000 Jan
2010 Jan
Month

TSLM Interpretation

- TSLM fits linear models to a time series and incorporates seasonality and trend data.
- The fitted line for the model created has a positive slope. This indicates a positive relationship between time and credits.

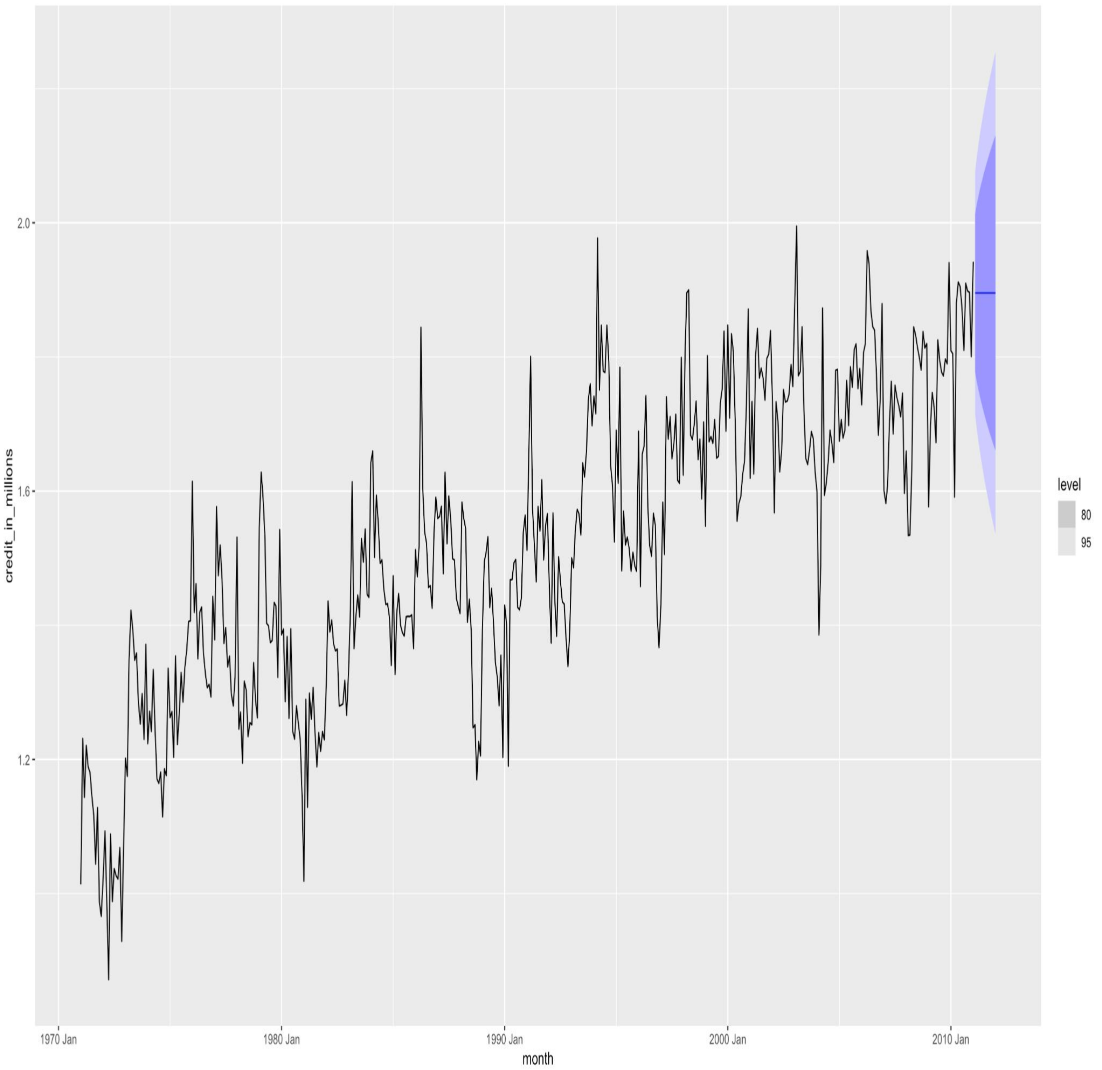
```
Series: ï..credit_in_millions
Model: TSLM

Residuals:
       Min         1Q     Median         3Q        Max
-0.3471317 -0.0735314  0.0004456  0.0719124  0.4090237

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.167e+00  1.095e-02  106.57   <2e-16 ***
month       4.546e-05  1.265e-06   35.95   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1213 on 490 degrees of freedom
Multiple R-squared: 0.7251,	Adjusted R-squared: 0.7245
F-statistic:  1292 on 1 and 490 DF, p-value: < 2.22e-16
```

- As 1 month increases in time results in a 0.00045 increase in credit (in millions).
- Time and Credits in millions are indicated to have a statistically significant relationship.

Exponential Smoothing Model

- Due to the low parameter you can see the trend here is staying steady.
- To be able to see if this is a useful ETS model, we would have to compare it to another ETS model.

AIC	AICc	BIC
676.4343	676.4846	688.9619



RMSE of our Models

Naive Model

	.model	.type	ME	RMSE	MAE	MPE	MAPE	MASE	RMSSE	ACF1
	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	Mean	Training	5.33e-17	0.208	0.169	-2.15	12.0	1.27	1.22	0.870
2	NaÃ¯ve	Training	1.89e- 3	0.102	0.0761	-0.106	5.20	0.570	0.601	-0.404
3	Seasonal naÃ¯ve	Training	1.61e- 2	0.170	0.133	0.489	9.10	1	1	0.630

Exponential Smoothing Model

	.model	.type	ME	RMSE	MAE	MPE	MAPE	MASE	RMSSE	ACF1
	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	ETS(ï..credit_in_millions)	Training	0.00315	0.0915	0.0695	-0.0624	4.64	0.532	0.553	0.00893

TSLM Model

.model	.type	ME	RMSE	MAE	MPE	MAPE	MASE	RMSSE	ACF1
<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
TSLM(ï..credit_in_millions ~ month)	Training	1.29e-17	0.120	0.0930	-0.712	6.39	0.713	0.725	0.643

ARIMA Model

.model	.type	ME	RMSE	MAE	MPE	MAPE	MASE	RMSSE	ACF1
<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
ARIMA()	Training	0.00898	0.0892	0.0675	0.302	4.49	0.517	0.539	-0.00625

Best Fit Model and Conclusion

- We compared the models using the RMSE
  - RMSE is the root mean squared error and this is just a fancy way of saying average distance between the predicted values from the model and the actual values in the dataset.
  - Therefore the smaller the RMSE the better fit and more accurate the model will be.
- That is why we found Arima to be our best fit model, that is because we running all of the RMSEs this was the smallest.
- Our smallest RMSE was 0.0892