Forecasting Methods

Handout

▼ Creating Models

- Note that when we are making use of the `fpp3` package, we will be creating model tables (`mable`) where it will store a table of the various models that we are working with
- Function for creating models: `model()`

▼ Making Forecasts

- Make use of the `forecast()` function and we will pass in the model table + the prediction horizon to make forecasts based on all the models that we have
- Note that we can put `show_gap=FALSE` to connect the forecast to the actual data
- Note that the `forecast()` can be used with multiple models in the `mable` object
- If there was any adjustment being done, the `forecast()` function will do the unbiased back transform

▼ Average Method

Takes the arithmetic average of all our observations

$$\hat{y}_{T+t|T} = \frac{y_1 + y_2 + \cdots + y_T}{T}$$

- h - Forecast horizon (The amount of time we want to forecast ahead)

▼ R Code

Note that we just need to specify `MEAN` as the model

```
mean_mtd <- insurance %>%
  model(avg = MEAN(Quotes))
```

▼ Naive Method

Make use of only the most recent observation as the prediction

$$\hat{y}_{T+h|T} = y_T$$

$$y_{T+h} = y_T + \sum_{k=1}^{h} e_{T+k}$$

Note that the Naive method is optimal when the data comes from a random walk.

▼ R Code

Note that we just need to specify `NAIVE` as the model

```
mean_mtd <- insurance %>%
  model(naive= NAIVE(Quotes))
```

▼ Random Walk with Drift

Includes a constant difference between successive observations apart from the random noise

$$y_t = \delta + y_{t-1} + e_t \qquad \text{for} \qquad t \geq 1$$

where δ is the constant difference

$e_t \sim WN(0, \sigma_e^2)$ and $y_0 = 0$

Value of δ

- For this naive approach, it will just be the sample variance between successive observations

$$\hat{\delta} = \frac{1}{T-1}\sum_{t=2}^{T}(y_t - y_{t-1}) = \frac{y_T - y_1}{T-1}$$

Point Forecast

$$\hat{y}_{T+h|T} = y_T + h(\frac{y_T - y_1}{T-1})$$

▼ **Derivations**

Note that at time $T+h$, we can express the value of y_{T+h} as

$$y_{T+h} = h\delta + y_T + \sum_{k=1}^{h} e_{T+k}$$

- Note that for the second part, it is a recursive relationship like what we have for the Naive method

Conditional Expectation is given by:

$$E(y_{T+h}|y_1, y_2, \cdots, y_T) = h\delta + y_T$$

- Note that for this, $h\delta$ and y_T are constant for the conditional expectation and each of the e_{T+k} has mean 0

▼ **Interpretation**

It is analogous to the naive forecast but with the extrapolated drift (number of forecast ahead we are making)

▼ **R Code**

Note that we just need to specify `RW` as the model and add the `drift()` function as part of the equation

```
aus_air_rwf <- aus_airpassengers %>%
  model(rwf = RW(Passengers ~ drift()))
```

Note that without the drift term, it is just a naive model

▼ Seasonal Naive Method

Sets each forecast to be equal to the last observed value form the same season of the previous year

$$\hat{y}_{T+h|T} = y_{T+h-km}$$

where $k = \lfloor \frac{h-1}{m} \rfloor + 1$

- Note that m is the period. k is essentially computing for the number of months we have to backtrack

▼ R Code

Note that we just need to specify `SNAIVE` as the model

```
aus_arrivals_sn <- filter(aus_arrivals, Origin == "Japan") %>%
  model(sn = SNAIVE(Arrivals))
```

▼ Residuals

▼ Innovation Residuals vs Response Residuals

- If we are applying a transformation $w = f(y)$ then:
  - $w_t - \hat{w}_t$ - Innovation Residuals
  - $y_t - \hat{y}_t$ - Response Residuals

- Note that we will analyze the innovation residuals
- If there is no transformation, **Innovation Residuals = Response Residuals**

▼ Essential Properties

1. Residuals should be uncorrelated
2. Residuals should have zero mean

▼ Desired (but not essential) Properties

1. Residuals have constant variance
2. Residuals are normally distributed

▼ R Code

`augment` can be used on a model object to extract the innovation residuals, fitted values and response residuals

- Note that we need to pass in the model object inside

`gg_tsresiduals` - can be used to study the residuals on a model object

- Note that we will be given the

▼ Tests for Autocorrelation

Note that for both of the test, the main idea is to take the sum of the squared residuals

▼ Box-Pierce Test

$$Q = T\sum_{k=1}^{h} r_k^2$$

- Note that h is the maximum lag observed and T is the number of observations

We make use of the squared values of the residuals

Interpretation

- If r_k is close to 0, then Q will be small → Means that the residuals look like white noise
- If r_k is large, then Q will be large → Means that there is autocorrelation between the residuals

▼ **Number of lags to use**

- Non seasonal data: $h = 10$
- Seasonal: $h = 2m$

Note that if $h \geq T/5$ then we will just take $T/5$ instead

$\min(h, \frac{T}{5})$

▼ **Ljung-Box test**

$$Q^* = T(T+2)\sum_{k=1}^{h}(T-k)^{-1}r_k^2$$

- T - Number of observations
- h - Maximum lag being observed

Interpretation

- Large values of Q^* suggests autocorrelation do not come from white noise

Null Hypothesis

$$\rho_k = 0 \quad \text{for all k}$$

- Note that under the Null hypothesis, both Q, Q^* have a χ^2 distribution with $(h-K)$ degrees of freedom
  - K - Number of parameters in the model (from which the residuals are being computed)
  - If it is based on raw data, $K=0$

▼ R Code

`portmanteau` - Feature in the feature_set of the features function

```
augment(mean_and_naive) %>%
  features(.innov, features=feature_set(tags="portmanteau"), lag=10)
```

▼ Analysis

- If the value is large, then there is strong evidence against the model being white noise (i.e. there is still autocorrelation between the residuals)
- If the p-value is small, same as above, we have strong evidence against the model being white noise

▼ Evaluating Forecasting Accuracy

Train-test split

▼ Residuals

Given by **Observed - Fitted**

- This makes use of the fitted model

▼ Forecasts Error

$$e_t = y_t - \hat{y}_{t|t-1}$$

where y_t is the test set data and $\hat{y}_{t|t-1}$ is the point forecast

- Note that for this, it is using the test set that we have held out
- Training Data: $\{y_1, y_2, \cdots, y_t\}$
- Testing Data: $\{y_{t+1}, y_{t+2}, \cdots\}$

▼ Scale-Dependent Errors

For this, we are computing the errors on the same scale as the data.

Disadvantage: We can't use it for comparison between series that involves different units

1. **Mean Absolute Error (MAE)**

$$\text{MAE} = \frac{1}{h}\sum_{k=1}^{h}|e_{T+k}|$$

2. **Root Mean Squared Error (RMSE)**

$$\text{RMSE} = \sqrt{\frac{1}{h}\sum_{k=1}^{h}|e_{T+k}^2|}$$

▼ Scale-Independent Errors

Can be used for comparison across time series with different units. We are mainly making use of percentage errors but they are **numerically unstable** (because we could be dividing by 0)**.**

It also penalizes negative values more.

$$y_{T+1} = 100,\ y_{T+2} = 50 \ \text{ and } \ \hat{y}_{T+1|T} = 50,\ \hat{y}_{T+2|T} = 100,$$

Then $p_{T+1} = 50\%$ but $p_{T+2} = -100\%$ although e_{T+1} and e_{T+2} have the same magnitude. Notice that, if $\{y_T\}$ is a non-negative series, then for an observed value, there is a maximum positive p_T but the negative p_T is unbounded.

Percentage Error: $100 \times \frac{e_{T+k}}{y_{T+k}}$

1. Mean Absolute Percentage Error (MAPE) (**Numerically unstable**)

$$\text{MAPE} = \frac{1}{h}\sum_{k=1}^{h} |p_{T+k}|$$

2. Symmetric MAPE (**Numerically unstable**)

$$\text{sMAPE} = \frac{1}{h}\sum_{k=1}^{h} 200\frac{(y_t - \hat{y}_t)}{(y_t + \hat{y}_t)}$$

3. Scaled Errors

▼ **For Non Seasonal Data**

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

e_j - Forecast errors from the testing data

$y_t - y_{t-1}$ - Residuals from the training data

▼ **For Seasonal Data**

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

Note that both the numerator and denominator are on the same scale and therefore it is scale independent.

$$\text{MASE} = \frac{1}{h}\sum_{k=1}^{h}|q_{t+k}|$$

1. Making use of the average error based on a naive error. Then we take our residuals for whatever model divided by the naive error.
2. This gives an indication of how well it does against the naive method and therefore allows comparison between different models

▼ **What to look out for**

- If the absolute value is < 1: Then it means that compared to the naive forecast, the residuals is much lower and therefore the model performed better
- If the absolute value is > 1: Then it means tha compared to the naive forecast, the residuals is much higher and therefore the model performed poorer

▼ **Cross Validation**

-

▼ Prediction Intervals

Provides an estimate for the probability that our point forecast will lie within. Expresses the uncertainty in the forecasts

$$\hat{y}_t \pm 1.96\hat{\sigma}$$

- where $\hat{\sigma_h}$ is an estimate of the standard deviation of the h - step forecast distribution

▼ **General way of writing**

$$\hat{y_{T+h|T}} \pm c\hat{\sigma}_h$$

where c is the multiplier on the coverage probability

Percentage	Multiplier
50	0.67
55	0.76
60	0.84
65	0.93
70	1.04
75	1.15
80	1.28
85	1.44
90	1.64
95	1.96
96	2.05
97	2.17
98	2.33
99	2.58

Nice reference table for the multipliers for different coverage of the prediction intervals that we want

Do we need to do a bias adjustment for the prediction intervals

▼ $\hat{\sigma}$ values for different benchmark methods

Note that $\hat{\sigma}$ is the residual standard deviation

1. **Mean Forecasts**

$$\hat{\sigma_h} = \hat{\sigma}\sqrt{1 + \frac{1}{T}}$$

▼ **Proof**

1. Note that the Mean model assumes that $E(y_t) = \mu, e_t \sim WN(0, \sigma^2)$

Point Forecast: $y_{T+h|T} = \bar{y}$ where $\bar{y} = \frac{1}{T}\sum_{t=1}^{T} y_t$

Note that $\bar{y}$ is an estimate of μ

The best estimate for the point forecast will be $\mu + e_{T+h}$

Where this can be estimated by $\bar{y} + e_{T+h}$

$$\begin{aligned} Var(\bar{y} + e_{T+h}) &= Var(\bar{y}) + Var(e_{T+h}) \\ &= \frac{\sigma^2}{T} + \sigma^2 \\ &= \sigma^2(1 + \frac{1}{T}) \end{aligned}$$

2. **Naive Forecasts**

$$\hat{\sigma_h} = \hat{\sigma}\sqrt{h}$$

Point Forecast: $y_{T+h} = y_T + \sum_{k=1}^{h} e_{T+k}$

- Note that for the point forecast for Naive is just the last observation

▼ **Proof**

$$\begin{aligned} Var(\hat{y}_{T+h|T}) &= Var(y_T + \sum_{k=1}^{h} e_{T+k}) \\ &= Var(\sum_{k=1}^{h} e_{T+k}) \\ &= h\sigma^2 \end{aligned}$$

3. **Seasonal Naive Forecasts**

$$\hat{\sigma_h} = \hat{\sigma}\sqrt{k+1}$$

where k is the integer part of $\frac{h-1}{m}$ and m is the seasonal period

Point Forecast: $y_{T+h|T} = y_{T+h-km}$
where $k = \lfloor \frac{h-1}{m} \rfloor + 1$

▼ **Proof**

$$\begin{aligned} Var(\hat{y}_{T+h|T}) &= Var(y_T + \sum_{i=1}^{k} e_{T+h-km}) \\ &= Var(y_T) + Var(\sum_{i=0}^{k} e_{T+h-km}) \\ &= 0 + (k+1)Var(e_{T+h-km}) \\ &= \sigma^2(k+1) \end{aligned}$$

4. **Drift Forecasts**

$$\hat{\sigma_h} = \hat{\sigma}\sqrt{h(1+\frac{h}{T-1})}$$

Point Forecast: $y_{T+h} = y_T + h(\frac{y_T - y_1}{T-1}) + e_{T+h}$

▼ **Proof**

$$\begin{aligned} Var(\hat{y}_{T+h|T}) &= Var(y_T + h(\frac{y_T - y_1}{T-1}) + e_{T+h}) \\ &= Var(y_T) + Var(h(\frac{y_T - y_1}{T-1})) + Var(e_{T+h}) \\ &= \end{aligned}$$

▼ **If the residual errors are not normally distributed**

We can do bootstrapping and sample from the residuals which only assumes that the forecast errors are uncorrelated

▼ **Procedure**

1. Compute the residuals from the training data (i.e. all the $y_t - \hat{y}_t$)
2. Sample randomly from the distribution of residuals e_t
3. For our point forecast, we will make use of $y_{T+1} = \hat{y}_{T+1|T} + e_{T+1}$ where e_{T+1} is the residual that was randomly sampled in step 2
4. We can continue doing 1-3 until we have gotten all the point forecasts for a particular simulation
5. We can then simulate steps 1-4 for many times so we can get many realizations of the time series
6. Once we have that, for each of the point forecast at each time point, we can create a distribution and compute the prediction intervals by calculating the percentiles for each forecast horizon.