ML for Finance: Linear Machine Learning Models

S. Yanki Kalfa

UCSD - Rady SOM

June 15, 2022

Outline

Recap of Univariate Models

- ARMA models require stationarity for stability
- We can check for unit roots using ADF tests
- The AR component of the ARMA allows us to forecast well into the future using the chain rule
- MA forecast become zero after the q^{th} order
- Forecasting financial time series is difficult
- We can select the lag length of the ARMA using Information Criterion: AIC or BIC

Multivariate ARMA: ARMAX

- What if we need more than just the history of the variable
- We can augment our ARIMA model with exogenous regressors
- Some examples:
  - Fama-French 3 factor model
  - Inflation
  - GDP
  - Stock Prices

ARMAX: Formulation

Formulation of the ARMAX is very simple:

$$y_t = \alpha + \sum_{p=1}^{P} \phi_p y_{t-p} + \beta X_{t-1} + \varepsilon_t + \sum_{q=1}^{Q} \theta_q \varepsilon_{t-q}$$

Where X_{t-1} is a vector of exogenous variables, and β is vector of coefficients.
The above formulation is an ARMAX(P,Q).
We assume stationarity.

ARMAX: Pros and Cons

- Pros:
  - Allows for additional regressors
  - In most cases augmenting the ARMA is useful
  - It is linear, easy to explain
  - You can forecast with it
- Cons:
  - Can get very large
  - Can only forecast 1 step ahead
  - You may need to try direct forecasting for more than 1 step ahead
  - Need to have exogenous variables to be stationary

ARMAX: Forecasting

Forecasting with the ARMAX is simple and follows directly from ARMA. Let's focus on the simple example of stationary ARMAX(2,2) with 1 exogenous variable.

Suppose the true model is given by:

$$y_{T+1} = \alpha + \phi_1 y_T + \phi_2 y_{T-1} + \beta x_T + \varepsilon_{T+1} + \theta_1 \varepsilon_T + \theta_2 \varepsilon_{T-1}$$

The forecast follows directly:

$$y_{T+1|T} = \alpha + \phi_1 y_T + \phi_2 y_{T-1} + \beta x_T + \theta_1 \varepsilon_T + \theta_2 \varepsilon_{T-1}$$

The MSE of the forecast:

$$\mathrm{E}[(y_{T+1} - y_{T+1|T})^2] = \mathrm{E}[\varepsilon_T^2] = 1, \text{ for } \varepsilon \sim WN(0,1)$$

ARMAX: Forecasting

What if we really want to forecast 2 steps ahead, then:

The true model is given by:

$$y_{T+2} = \alpha + \phi_1 y_{T+1} + \phi_2 y_T + \beta x_{T+1} + \varepsilon_{T+2} + \theta_1 \varepsilon_{T+1} + \theta_2 \varepsilon_T$$

$$y_{T+2|T} = \alpha + \phi_1 y_{T+1|T} + \phi_2 y_T + \beta \underbrace{x_{T+1}}_{\text{We do not know this value}} + + \theta_2 \varepsilon_T$$

We are left with:

$$y_{T+2|T} = \alpha + \phi_1 y_{T+1|T} + \phi_2 y_T + \theta_2 \varepsilon_T \implies \text{ARMA Forecast}$$

The forecast is clearly misspecified.
I do not suggest forecasting 2 steps ahead. The time series literature uses Vector Autoregression (VAR) to get around this problem.

ARMAX: Solution

- So, is ARMAX useless?
- No, we can use recursive forecasting
- In finance, normally we do not forecast more than 1 period ahead
- If you really want to forecast using X, then endogenize it

ARMAX: Solution

- So, is ARMAX useless?
- No, we can use recursive forecasting
- In finance, normally we do not forecast more than 1 period ahead
- If you really want to forecast using X, then endogenize it

ARMAX: Recap

- ARMAX allows for exogenous variables
- The number of exogenous variables can be large
- Estimation error can be very large
- ARMAX does not perform model selection
- Can only forecast 1 step ahead

Loss Functions: Introduction

- Loss functions show the preference of the policy maker or forecaster's cost of forecast errors
- Trade-offs between forecast errors are quantified by the loss function
- We may prefer overpredicting to underpredicting
- So, the cost of negative forecast errors are smaller realtive to positive forecast errors
- Forecasts feed into policy making
- Examples:
  - Central Banks (Inflation, Unemployment, GDP)
  - IMF
  - World Bank

Loss Functions: Notation

- Outcome: Y
- Information set X
- Forecast: $f = f(X)$
- Forecast error: $e = Y - f$
- Loss function: $L(f, y) \to \mathbb{R}$

How to pick a Loss Function

- The main purpose of a Loss Function is to weight the cost of forecast errors
- The choice of Loss Function has an effect on
  - forecasting models
  - estimated parameters
  - forecast comparison

Assumptions on Loss

There are three main assumptions that a Loss Function needs to satisfy

Assumption

$L(0) = 0$: *Normalization*

$L(e) \geq 0$ *for* $|e| > 0$: *Imperfect forecasts generate larger loss than perfect ones.*

$L(e)$ *is monotonically non-decreasing in* $|e|$

- $L(e_1) \geq L(e_2)$ *if* $e_1 > e_2 > 0$
- $L(e_1) \geq L(e_2)$ *if* $e_1 < e_2 < 0$

Common Loss Functions: MSE

The Mean Squared Error loss function is the most widely used loss function. Refer to as MSE Loss.

$$L(f, y) = \alpha(y_t - \hat{y}_t)^2 \mid \alpha > 0 \tag{1}$$

- It satisfies Assumptions 1 - 3
- It is symmetric
- Differentiable everywhere
- Convex: large errors are penalized at an increasing rate

$$\hat{y}_t^\star = \arg\min_{\hat{y}} \mathrm{E}[(y_t - \hat{y}_t)^2]$$

$$FOC:$$

$$\hat{y}_t^\star = \mathrm{E}[y_t]$$

This aligns perfectly with OLS.

Common Loss Functions: lin-lin Loss

The Piece-wise linear (lin-lin) loss is a nice way representing different preferences for underpredicting vs. overpredicting.

$$L(e) = (1-\alpha)e1_{e>0} - \alpha e1_{e\leq 0} \mid 0 < \alpha < 1 \tag{2}$$

$$e1_{e>0} = \begin{cases} 1 & \text{for } e > 0 \\ 0 & \text{Otherwise} \end{cases}$$

$$e1_{e\leq 0} = \begin{cases} 1 & \text{for } e \leq 0 \\ 0 & \text{Otherwise} \end{cases}$$

- It satisfies Assumptions 1 - 3
- Differentiable everywhere except at zero
- Weight on overpredicting: $(1-\alpha)$
- Weight on underpredicting: α
- Mean Absolute Error if $\alpha = 1/2$

lin-lin Loss: Optimal Forecast

Risk under lin-lin loss is defined as:

$$\mathrm{E}_Y[L(Y - \hat{Y})] = (1-\alpha)\mathrm{E}[Y|Y > \hat{Y}] - \alpha\mathrm{E}[Y|Y \leq \hat{Y}]$$
$$FOC:$$
$$\hat{Y}^{\star} = F_Y^{-1}(1-\alpha)$$

- Optimal forecast is the $(a - \alpha)$ quantile of Y
- If $\alpha = 1/2$ then the median is the optimal forecast
- As $\alpha \to 1$ overpredicting becomes more costly

lin-lin Loss: Optimal Forecast

Risk under lin-lin loss is defined as:

$$\mathrm{E}_Y[L(Y - \hat{Y})] = (1-\alpha)\mathrm{E}[Y|Y > \hat{Y}] - \alpha\mathrm{E}[Y|Y \leq \hat{Y}]$$
$$FOC:$$
$$\hat{Y}^{\star} = F_Y^{-1}(1-\alpha)$$

- Optimal forecast is the $(a - \alpha)$ quantile of Y
- If $\alpha = 1/2$ then the median is the optimal forecast
- As $\alpha \rightarrow 1$ overpredicting becomes more costly

Linex Loss

The linear exponential loss is another assymetric loss function.

$$L(e) = \alpha_1(\exp(\alpha_2 e) - \alpha_2 e - 1) \mid \alpha_2 \neq 0,\ \alpha_1 > 0 \tag{3}$$

- Differentiable everywhere
- α_2 controls direction and degree of asymmetry
- $\alpha_2 > 0$ L(e) is linear for overpredicting and exponential for underpredicting
- underpredictions is more costly than overprediction

Case for Asymmetric Loss Functions

- Decision makers may have different preferences
- It is more costly to overpredict GDP than underpredicting
- This may not be the case for inflation
- If underpredicting is more costly then it might be optimal to overpredict (negative bias)
- If overpredicting is more costly then it might be optimal to underpredict (positive bias)

Model Selection: Motivation

- Started with ARMA models
- Added exogenous variables
- Number of exogenous variables can be large
- Certain variables could be weak predictors
- How de we select the variables we want to include?

Model Selection: Introduction

- We normally have more than 1 model to forecast
- The models can vary
  - Number of lags
  - Number of predictor variables
  - Parametric vs. Non-Parametric
- The goal is to come up with the best model
- There could be models with similar performances
- The relationship between y and X can change

Model Selection: Notation

- Let $\mathbb{M}_K$ be the set of models
- $M_k \in \mathbb{M}_K$ for $k = 1, 2, \ldots, K$
- Let X_t be the space of predictors
- β_k is a vector of parameters for M_k
- We search over $\mathbb{M}_K$ to find the best model(s).

Model Selection: Notation

- Let $\mathbb{M}_K$ be the set of models
- $M_k \in \mathbb{M}_K$ for $k = 1, 2, \ldots, K$
- Let X_t be the space of predictors
- β_k is a vector of parameters for M_k
- We search over $\mathbb{M}_K$ to find the best model(s).

In Sample Fit vs. Forecasting

- In all cases larger models that nest smaller models produce better in sample fit, think of R^2.
- In general, larger models do not outperform smaller models when we look at forecasting performance.
- We should not be tempted by large complex models when forecasting. Parsimony is beautiful.
- When doing model selection, we need to think of two potential problems, and their trade-off.
- Large complex models may have small misspecification error and large estimation error
- Small models may have large misspecification error (omitted variables, functional form) but have small estimation error.

Going from Large to Small Models

- How can we go from large to small models
- Answer: Machine Learning
- Sequential Feature Selection
- LASSO
- Ridge
- Elastic Net

Going from Large to Small Models

All models listed above start with the kitchen sink models. Meaning they start with all possible predictors in the feature space.

$$y_t = \alpha + \sum_{p=1}^{P} \beta_p x_{p,t-1} + \varepsilon_t \tag{4}$$

- Sequential Feature Selection uses t-stats to pick the best model. This could be done either forward or backward
- Penalized Regression: augments the loss function by penalty to pick best variables or shrink the magnitude of the variables
  - LASSO: L1 penalty
  - Ridge: L2 penalty
  - Elastic Net: L1 and L2 combination

Sequential Feature Selection (SFS)

- SFS selects the best subset of regressors to include in the regression
- It is done by sequentially testing the coefficients of the variables included
- Widely used in financial forecasting

SFS- Backward Selection- General to Specific

- We start from equation 4
- Rank variables by t-scores
- Eliminate the variable with the smallest t-score below some threshold
- Re-run regression and repeat the process until some threshold is reached

$$t_{min} = \min_{k=1,\ldots,K} |t_{\hat{\beta}_k}| < \underline{\mathrm{t}}$$

A rule of thumb is setting $\underline{\mathrm{t}} = 2$.

SFS- Forward Selection- Specific to General

$$y_{t+1} = \alpha + \varepsilon_{t+1} \quad (5)$$

- We start from equation 5
- Test each variable one by one
- Rank the variables by t-scores
- Add the variable with highest t-score above some threshold
- Re-run regression and repeat the process until some threshold is reached

$$t_{max} = \max_{k=1,\dots,K} |t_{\hat{\beta}_k}| > \bar{t}$$

A rule of thumb is setting $\bar{t} = 2$.

Pros and Cons of SFS

- Pros:
  - Easy to interpret
  - Easy computation
  - Simple
- Cons:
  - Selection of variables is path dependent
  - Cannot search over all possible models
  - Cannot really control the size of the model

Penalized Regressions

- Up until we considered the MSE Loss (Equation (1))
- Is there a way of augmenting the MSE Loss to also achieve model selection
- Yes
    - LASSO
    - Ridge
    - Elastic Net
- Minimize MSE subject to some type of penalty
- Goal is to set some parameters to zero or shrink all parameters
- This controls for overfitting

Penalized Regressions: LASSO

The Least Absolute Shrinkage and Selection Operator (LASSO) augments the MSE with an L1 penalty term as described below:

$$\mathcal{L}(\beta) = \frac{1}{T}\sum_{t}^{T}\left(y_t - \beta' X_{t-1}\right)^2 + \lambda \sum_{p=1}^{P} |\beta_p| \tag{6}$$

The optimal β solves the below function

$$\hat{\beta}^{LASSO} = \underset{\beta}{\arg\min}\, \frac{1}{T}\sum_{t}^{T}\left(y_t - \beta' X_{t-1}\right)^2 + \lambda \sum_{p=1}^{P} |\beta_p| \tag{7}$$

Penalized Regressions: LASSO

- There are no analytical solution to equation (7)
- We solve it with numerical optimization
- λ is a tuning parameter, a pure LASSO sets $\lambda = 1$
- As λ grows $\hat{\beta}$ goes to zero
- Works really well for sparse models

Penalized Regressions: Ridge

The Ridge regression augments the MSE with an L2 penalty term as described below:

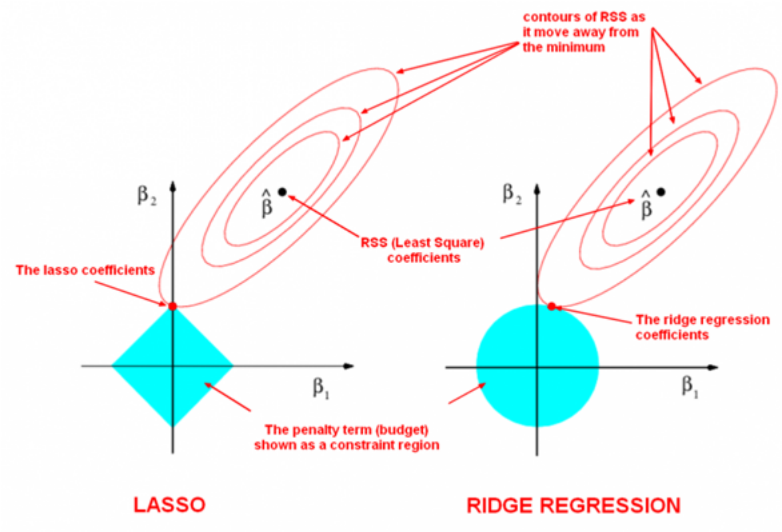
$$\mathcal{L}(\beta) = \frac{1}{T}\sum_{t}^{T}\left(y_t - \beta' X_{t-1}\right)^2 + \lambda \sum_{p=1}^{P} \beta_p^2 \tag{8}$$

The optimal β solves the below function

$$\hat{\beta}^{Ridge} = \arg\min_{\beta} \frac{1}{T}\sum_{t}^{T}\left(y_t - \beta' X_{t-1}\right)^2 + \lambda \sum_{p=1}^{P} \beta_p^2 \tag{9}$$

- Analytical solution to equation (9)
- λ is a tuning parameter, a pure Ridge sets $\lambda = 1$
- As λ grows $\hat{\beta}$ gets closer to zero but never reaches
- It does not perform any model selection, it only penalizes very large

Ridge and LASSO: Geometric Interpretation



Penalized Regression: Elastic Net

The Elastic Net combines both LASSO and Ridge penalties. The loss function is given by the below function.

$$\mathcal{L}(\beta) = \frac{1}{T}\sum_{t}^{T}\left(y_t - \beta' X_{t-1})\right)^2 + \lambda(1-\alpha)\sum_{p=1}^{P}|\beta_p| + \frac{\lambda\alpha}{2}\sum_{p=1}^{P}\beta_p^2 \quad (10)$$

The optimal β solves the below function

$$\hat{\beta} = \underset{\beta}{\arg\min}\, \frac{1}{T}\sum_{t}^{T}\left(y_t - \beta' X_{t-1})\right)^2 + \lambda(1-\alpha)\sum_{p=1}^{P}|\beta_p| + \frac{\lambda\alpha}{2}\sum_{p=1}^{P}\beta_p^2 \quad (11)$$

Penalized Regressions: Elastic Net

- No analytical solution to equation (11)
- α is the weight attributed to the L1 and L2 penalties. We set $\alpha = 1/2$, but can be tuned using Cross Validation.
- λ is still a hyperparameter. We can tune it with CV. We can also just let $\lambda = 1$
- It performs selection and shrinkage