BEMM458 Programming for Business Analytics

December 2023

1 Introduction

The data for this assignment is comprised of a number of US economic time series originally from the FRED-MD database: here and the NBER here. See the links for more information on the data used in the assignment.

To complete your assignment, you should submit:

- Two python scripts: `QuestionA.py, QuestionB.py`
- A notebook file `QuestionC.ipynb`

These scripts will be run and your marks will depend on the completeness and accuracy of the results produced by your code. DO NOT use hard coded path variables in file import/output commands i.e. your code should adopt standard Python behaviour and read/write all files to the 'current working directory'.

ANSWER ALL QUESTIONS - PAPER TOTAL 120 MARKS

2 Question A - Time Series Forecasting

The task is to fit simple time series models to the data, and calculate their predictive ability. This is done by via a recursive forecasting exercise. That is, for each 'backtest' month in the sample, we fit a model to the *training* data at that point in time, and predict each variable for a h step ahead *test* period. We then measure the predictive ability of our model on the complete set of test data.

We will fit three different exponential smoothing models to each time series as at each backtest month. These models are documented in Rob Hyndman's online forecasting textbook. We will use the Python Statsmodels implementation of these models here.

We fit three models:

1. Model $'A'$: Simple Exponential Smoothing (`ETSModel` with default parameters)
2. Model $'AA'$: Exponential Smoothing with additive trend (`ETSModel` with trend=$'add'$)
3. Model $'AAd'$: Exponential Smoothing with damped additive trend (`ETSModel` with trend=$'add'$ and $damped\_trend = True$)

At each backtest date, once all models are fitted we will choose the best model using an in sample fitting criteria known as the 'Corrected Akaike Information Coefficient' (see Hyndman's text for further details). We then forecast each series 12 months ahead using the chosen model. The function to fit the models is written for you and contained in the file `'FRED_MD_Tools.py'`. For a given time series, this function returns the code ('A','AA','AAd' as above) for the selected model, and h period ahead forecasts. We will calculate forecast accuracy using a slightly modified version of the 'Out of Sample R^2' statistic [1]

Use the template `'QuestionA.py'` as a base for your answer.

1. Write a function called `load_data()` to import the FRED-MD data from file in to a Pandas DataFrame. The data is formatted with a set of 'Transformation' values as the first row of data. These need to be removed. Format the data index as a monthly 'Period Index'. Give your function an appropriate docstring. The column names in this dataframe represent series identifier codes. The function should return the formatted data frame.

[1] (Campbell, John Y., and Samuel B. Thompson. 2007. "Predicting Excess Stock Returns Out of Sample: Can Anything Beat the Historical Average?" The Review of Financial Studies 21 (4): 1509–31)

[3 marks]

2. Using your function, load and format the `'2023-09.csv'` FRED-MD file. Store your formatted DataFrame in the the variable `'FRED_DATA'`.

[4 marks]

3. Write Python code to loop through each backtest date in `backtest_dates`. This variable is initialised for you in the template.

   - For each backtest date, fit the models to each series in the database as at that date.
   - You will notice that some series have missing values. To select valid data, call the predefined function `select_continuous(srs)`, found in `FRED_MD_Tools.py`. This will return a continuous set of data from *srs* which can be used as training data and passed in to the model fitting function.
   - Fit the models using the function `fit_models`. Capture for each model the set of $h = 1...12$ step ahead forecasts.
   - Record in a data frame (`'models'`) which model ('A','AA','AAd') was chosen for each series identifier for the given backtest date.
   - Record also for each series identifier / backtest date the $'naive'$ forecast - $\tilde{y}$ - defined as the last available value of the series before the forecast was made. This corresponds to the optimal forecast if each series follows a random walk - i.e. the series is not at at all predictable.
   - Calculate the forecast errors $(y_t - \hat{y}_t)$ for EACH ETS model and for the naive model at EACH horizon h.
   - From the vectors of 12 step ahead forecasts (model and naive), select for each series identifier at each backtest date the 3, 6, 9, and 12 step ahead forecasts (these values are set up as a list in the template file.) and store in dataframes named as described below. You should have 9 dataframes as follows (check carefully that each have variable names as defined below):
     - 4 dataframes with variable names `'horizon_3', 'horizon_6','horizon_9','horizon_12'` with an index corresponding to `'backtest_dates'`, and columns corresponding to the full set of series identifiers. Each will contain the h step ahead forecast error $(y_t - \tilde{y}_{t,h})$ as at $h = [3, 6, 9, 12]$.
     - The dataframe (variable name `'models'`) with the set of models chosen (as at the last backtest date), with index equal to the set of identifiers.
     - 4 dataframes (variable name `'naive_3', 'naive_6','naive_9','naive_12'`) with an index corresponding to `'backtest_dates'`, and columns corresponding to the full set of series identifiers. These dataframes hold the naive forecast errors $(y_t - \tilde{y}_t)$ as at each backtest date.

[25 marks]

4. Calculate for each $h = [3, 6, 9, 12]$ the out of sample R^2 statistic:

$$R^2 = 1 - \frac{\sum_{t=1}^{T}(y_t - \hat{y}_t)^2}{\sum_{t=1}^{T}(y_t - \tilde{y}_t)^2}$$

   where $\hat{y}$ is the h step ahead ETS forecast and $\tilde{y}$ is the naive forecast, store these in 4 dataframes `'R2_3M','R2_6M','R2_9M','R2_12M'`.

[4 marks]

5. Create a dataframe with the series identifiers as an index, and 2 columns [`'model'` and `'R2'`]. Store the latest model selected, and the 3 month R^2 statistic in this datafame and save as a CSV file `'fcast_res.csv'`.

[4 marks]

3 Question B - Recession Prediction

The task is to build classification models to predict recessions using the FRED-MD Data. We use a supervised learning approach, with months corresponding to recessions identified by a panel of experts (and with the benefit of hindsight) supplied as the NBER data mentioned above. This data classifies each month as 'Recession' or 'Expansion'. We run a recursive forecasting exercise, attempting to predict the classification of each month in to 'recession' or 'expansion' 6 months ahead. In order to do this we initially compress the data from 127 seperate series down to 8 'indicies' using Principal Components Analysis (PCA). Functions to compute the Principal Components, and fit the classification models are pre-written for you and can be found in `'FRED_MD_Tools.py'`. The PCA uses transformed FRED-MD data as described in the Mcracken & Ng paper on FRED-MD. here. This transformed data is pre-computed for you. We will use Logistic Regression and Support Vector Classifier from the SciKit Learn library to fit the models.

Use the template file `'QuestionB.py'` to complete your answer.

1. Import the files `'2023-09-TF.csv'` and `'NBER_DATES.csv'`, and format the indices appropriately.

[4 marks]

2. Set up a dataframe `'prob_results'` to hold the probabilities of recession obtained for the models. The index of this data frame should correspond to `'backtest_dates'` (pre-specified in the template file). The columns of this dataframe should be set to `['LogisticRegression','SVC']`.

[3 marks]

3. At each backtest date t:

   - Select the full set of time series data available at time t
   - Remove from this any time series where there are less than 36 non-missing values.
   - Standardise each series (by subtracting the series mean and dividing by its standard deviation). Fill any remaining missing values with zeros.
   - Compute, using this data 8 Principal Components (PCs) using `'pca_function()'`.
   - Shift the recession indicator data back 6 months (we are predicting 6 months ahead, so the PCs from time t are use to predict the recession/expansion classification for time $t+6$), and select the appropriate values for each month in D_t
   - Use the supplied function `'fit_class_models()'` to fit the classification models. This function returns the probabilities of expansion/recession and the Scikit Learn model objects for both models.
   - Store the probabilities of recession in `'prob_results'` for each backtest date.

[18 marks]

4. When the procedure has run for each backtest date, select the actual NBER classifications for each backtest date. Remember to shift the NBER data back 6 months to do this. Forecast accuracy for binary outcomes can be assessed using the Brier Score here (lower is better). This is calculated as:

$$BS = \frac{\sum_{t=1}^{T}(f_t - o_t)^2}{T}$$

   Where f_t is the model probability of the event happening, and o_t is 1 if the event (recession) occurs, and 0 otherwise (expansion). Calculate the Brier Scores for each model over the set of backtest dates and store in variables `'brier_lr', 'brier_svc'`.

[9 marks]

5. Calculate a 'Combination' model, as the average of the recession probability for both models, store the probabilities for this model in `'prob_results['Comb']'`. Calculate a brier score for this model as **'brier_comb'**.

[4 marks]

6. Select the best predictive model (Logisitc Regression, Support Vector Classifier or Combination) according to the Brier score. Select the corresponding values and write these to a csv file `'RecessionIndicator.csv'`.

[2 marks]

4 Question C - Forecasting Financial Time Series

The FREDMD Appendix file contains meta data on each of the time series in the database. Included in this table is a 'group' variable, which collects together similar economic time series as defined in the Mckracken & Ng Working paper here. Group 8 corresponds to Financial Variables. In this question you are asked to explore the predictive power of models fitted in Question B for these variables.

Submit your answer to this question as an Python Notebook ('.ipynb') file. Include text and working code to complete your analysis, illustrate your answer with visualisations produced within the notebook. If you use any code libraries which are not part of the standard Anaconda Library, please note and describe these, specifying why you have used them, before you execute any `import` statements.

1. Import the recession probabilities for your chosen predictive model calculated in Question B above. Does this model have predictive power for the set of Financial Variables? Using Statsmodels OLS, Estimate a regression model for each financial variable (taken from the 'Transformed' data set used in Question B) on the recession probability estimated from Question B (which you saved in the file `'RecessionIndicator.csv'`) lagged by 1 month.

[10 marks]

2. Compare your results to a simple auto regressive model of order 1 (i.e. regress each variable on its first lag).

[10 marks]

3. Discuss the predictive power of these models for each of the Financial Variables, illustrating with charts where appropriate.

[10 marks]

4. Comment on how you approached the three questions in this assignment. Document in particular if and how you utilised AI based coding tools to develop any of your code, which tools you have used, and how you developed the prompts used to generate code. Show the exact prompt which generated any code you have used. Comment on the efficiency / accuracy of any code generated in this way, and describe any modifications you made. Note any part of the code you are submitting for this assignment which was generated by AI tools.

[10 marks]

TOTAL 120 MARKS - END OF PAPER