Machine Learning:

How high are the recession probabilities in the United States?

Edoardo Pilla

ID: 7744044

ABSTRACT

The paper utilizes a support vector machine and a neural network to predict the likelihood of recession in the United States over a twelve-month horizon, in a binary classification framework, confirming the importance of term spreads as predictors for periods of economic decline, and highlighting the better performance of neural networks by means of the main evaluation metrics.

Additionally, the model from “The Near-Term Forward Yield Spread as a Leading Indicator: A Less Distorted Mirror”, written by Eric Engstrom and Steve Sharpe in Finance and Economics Discussions Series 2018-055 within the Board of Governors of Federal Reserve System, is replicated.

1. Introduction

Recessions are defined by the National Bureau of Economic Research as periods of time between economic activity peaks and troughs, characterized by sensible decline in economic activity across different sectors, lasting for multiple months. Compared to expansions, their specular counterparts, they are usually shorter and more pronounced, albeit less frequently occurring.

Knowing in advance when a recession might happen is critical for portfolio managers and practitioners within the financial industry, because recessions are linked to downturns in stock markets, especially in their initial phase, hence exposure to risky assets may be adjusted accordingly, to avoid suffering important drawdowns.

Machine learning methods have been successfully implemented across a wide array of different applications, ranging from computer vision to natural language processing, and finance is but one of the many fields where such approaches can be used, due to the availability of high-quality data, which are strongly demanded by such models.

This essay aims at employing information about term spreads to provide a forecast of recession probabilities in the United States at a twelve-month horizon, within a binary classification framework, using a support vector machine and a neural network, so that the output is readily interpretable by asking the question: “Are recession probabilities going to be high or low in the next twelve months?”

The rest of the paper is structured as follows: after the literature review, in which previous evidence about the usage of term spreads as predictors for future recessions is collected, along with results from other studies applying machine learning models to the same task, the methodology is discussed in detail with a particular focus on the employed data, and the baseline models are extended to select a richer feature set, finally summarizing the results. A section is then dedicated to the replication attempt of the paper mentioned in the abstract, and eventually conclusions are drawn. Two appendices wrap up with respect to data sources and Python code.

1. Literature review

Ample literature exists which ties the yield curve and its behavior over time with future changes in economic activity. Following the chronological order of publications, among the first researchers to discuss the topic, Estrella and Hardouvelis (1991) conclude that its slope can predict changes in real output and offers additional predictive power over several different variables, including lagged output growth and inflation in the United States.

Bernard and Gerlach (1998) extend the study by showing that the yield curve is not only able to forecast recessions in the United States, but also in seven more countries and up to seven quarters ahead. They also state that the predictive power is higher in some nations compared to others, most likely because of differences with respect to financial regulations, leading to a suboptimal reflection of market expectations, and draw attention to the fact that most recessions followed attempts by the respective central banks to tackle inflation.

Chauvet and Potter (2005) add up to the literature by comparing four probit model specifications to gradually consider breakpoints within the business cycle and serially correlated errors. They warn of the tendency in baseline models to overpredict recessions, highlighting the fact that if no further explanatory variables, other than the term spreads, are included into the specification, then it follows that those will be the only sources of information deemed relevant by the model itself.

Wright (2006) also employs different specifications of probit models to forecast recessions, and advocates making use of the level variables for the interest rates, in addition to the term spreads, in order to achieve a better in sample fit and predictive performance, concluding that a flat yield curve is not necessarily linked to a subsequent economic downturn, an element which is likely missed by specifications relying only on term spreads. He also notes that downward sloping yield curves have been reflecting, for example in Australia and the United Kingdom, factors different from a tight monetary policy, and subsequent economic expansions had been observed.

Piazzesi et al. (2006) aim at forecasting GDP using a model based on yields in the United States, a task closely related to the one the paper deals with, and find that the best measure of slope for the yield curve is given by the maximal maturity difference, while the nominal short rate does a much better job at forecasting GDP, compared to the yield slope, both in and out of sample.

Rudebusch and Williams (2008) focus on showing how simple models depending on real-time information coming from the yield curve have significant predictive power, and provide more reliable forecasts several quarters ahead, than those generated by macroeconomic professionals, once again proving how market information about interest rate spreads can be effectively applied for recession prediction tasks.

Chinn and Kucko (2015) find, on a slightly different note, that the predictive power of the yield curve decreased in the progression of the Great Moderation, but increased again during and after the Global Financial Crisis of 2008, by examining its performance as a predictor for economic activity both in the United States and in a set of developed European countries, where models actually perform better when considering more recent data, but they also signal how including information about short term interest rates in levels widens the performance gap across countries, with Japan and Italy revealing the worst prediction outcomes, leading them to identify the United States as an outlier, in terms of how the yield curve can be used to predict economic activity.

Gogas et al. (2015) extend the literature on the usage of information about the arc of the yield slope by forecasting recession instances using quarterly data about seasonally adjusted GDP and a range of different interest rates on government bonds from 1976 to 2011 in the United States, comparing the performance of the baseline probit model with different specifications of support vector machines, based on several types of kernels, allowing them to predict the output variable without any a priori assumption. They claim that long term interest rates have no significant informative power over short term ones because the United States are expected to have a stable long term economic outlook, and argue that using the radial basis function kernel yields the best performance, by means of all the evaluation metrics designed for this kind of task, namely accuracy, recall and F score.

Engstrom and Sharpe (2018) further develop the previously outlined point by arguing that the usual choices for term spreads, such as the one formed by the difference between the 10-year Treasury bond and the 2-year counterpart, are worse off than what the authors define as a “near-term forward spread”, which measures market expectations with respect to the trajectory of monetary policy, hence they conclude that relying on spreads formed by taking into account yields with maturities longer than 2 years does not offer any additional advantage in predicting recessions in the United States, especially at a one-year horizons.

Puglia and Tucker (2020) display contrasting evidence that simpler models such as the baseline probit regression should be preferred to most of the machine learning approaches, including, but not limited to, support vector machines and neural networks, if a more conservative cross validation procedure is carried out instead of the classical k-fold one, because the latter biases accuracy measures optimistically. In their study, the authors also account for more financial and macroeconomic indicators, for example the Conference Board’s Leading Economic Index, or the Aruoba-Diebold-Scotti business conditions index, which are acknowledged among the predictors for recessions in the United States, and in doing so, they highlight how machine learning techniques enable to capture features of the joint empirical distribution of interest rates, an opportunity out of the baseline probit model scope.

Ajello et al. (2022) build up on the work of Engstrom and Sharpe (2018), who assume that the near-term forward spread (NTFS), defined as the difference between the implied forward rate on 3-month Treasury bills, 18-months ahead, and the current rate on 3-month Treasury bills, proxies market sentiment with respect to monetary policy stance and short-term inflation expectations well, and thus can serve as a valuable leading indicator of economic activity, preferable to the respective long term spread, which in the view of Engstrom and Sharpe (2018) decreases the signaling content provided by forward rates.

Lastly, Kiley (2023) emphasizes that models whose only explanatory variables are term spreads will tend to overestimate the recession signals generated by the term spreads themselves, consistently with Chauvet and Potter (2005), because they are highly procyclical and proxy for indicators of the business cycle state, such as inflation and unemployment rates, which in turn can be effective predictors for recessions over the medium term, partially in line with what was already claimed by Estrella and Hardouvelis (1991).

1. Methodology

All the data used to develop and implement the baseline support vector machine and neural network have been provided by the Federal Reserve Economic Data website and are initially implemented at a monthly frequency from January 1982 to March 2023, to coherently estimate over a 12-month horizon. Macroeconomic variables included in the extended specifications, namely the Industrial Production Index and the Total Nonfarm Payroll Employment, were extracted from the FRED-MD, originally developed by McCracken and Ng (2015), which has recently become the standard for research employing macroeconomic data because it is updated in real time, and actively handles data changes and revisions.

In order to output a binary outcome as a result of the model, interpretable as a probability range, the dummy variable time series, developed by FRED using NBER data was extracted, where recessions are labelled as 1, while expansions are labelled as 0. This is in line with what Berge and Jordà (2011) suggest, since they find no significant evidence to prefer other measures of recession, such as the ones provided by Chauvet and Potter (2005) and Chauvet and Piger (2008), to the more immediate one used here. It is relatively safe to assume that the output series is reliable at the current time of analysis, as NBER states that the Business Cycle Dating Committee has never changed any cycle date since its inception in 1978, with the last trough date identified as April 2020, and reported in July 2021.

Despite the more recent evidence collected by Engstrom and Sharpe (2018) and Ajello et al. (2022), who encourage focusing on forward term spreads below the 2-year mark for forecasting purposes at short and medium horizons, the initial choice of input features for the model results in the 10-year to 3-month spread between Treasury bills, found by Choi et al. (2023), using machine learning techniques, to be the most appropriate for predictive purposes, along with the 10-year to 2-year spread between the same class of securities, another commonly chosen maturity pair in the literature, together with the 10-year to 1-year, for example by Bauer and Mertens (2018). This is done to retain the clear interpretability of term spreads between yields and to identify their role as predictors for recessions, which represent the main purposes of the essay.

A frequently occurring problem when the main goal is forecasting in a time series context is the so-called look ahead bias, well addressed by Banz and Breen (1986), who compare the performance of portfolios based on accounting measures released in COMPUSTAT, before and after correcting for this error. The bias is characterized by the misleading ex-post assumption that a specific data point was already available as part of the information set, at a certain point in time, even if that same data point would have only been publicly reported at a later stage due, for example, to computational delays. Since term spreads are market-based data, they are immediately accessible in real time, hence they do not pose any look ahead bias threat in the original specification, whose only input features are term spreads themselves, although this issue will become relevant when the set of predictors will be extended to include macroeconomic variables. In fact, data about the Industrial Production Index for a given month are published halfway through the subsequent month, whereas data about Total Nonfarm Payroll Employment for a given month are published on the first Friday within the subsequent month. In practice, this should hint at correctly aligning input features when constructing the predictor matrix, considering that in the essay the forecast is performed at the end of the month, even if the dates within the datasets are indexed at the start of the month for improved readability.

The paper by Puglia and Tucker (2020) serves as a starting point for the implementation of the support vector machine used to carry out the 12-month ahead forecast. The authors modify the output series by setting the value to 1 if, at any of the following 12 months, a recession occurred as defined by NBER, and to 0 otherwise, almost doubling the number of true values compared to the original series. This both acts as a way of providing a forward-looking view to the algorithm, and increases the balance between the two classes within the dataset, since recessions represent a relatively rare occurrence.

An important remark that Puglia and Tucker (2020) make is that the common cross validation methods, which are normally employed when applying a machine learning algorithm to a cross sectional dataset, are not suitable for time series data, where serial correlation violates the i.i.d. assumption, while methods such as k-fold cross validation would lead to the so-called data peeking bias, resulting in optimistically skewed estimation performance. A simplified version of the nested time series cross validation approach that the authors implement as an alternative is proposed in the paper, which also respects the time ordering component characterizing time series data.

A grid search was performed to find the best hyperparameters during the training phase, and differently from what the authors report, the choice of hyperparameters was extended to include the kernel and its relative degree when polynomial, the cost factor C, which intuitively functions as a regularization parameter that, if large, prioritizes any hyperplane achieving less misclassifications, taking the support vector machine closer to its hard-margin definition, and the gamma value, which forces the decision boundary to be more linear if small, while the opposite holds if the hyperparameter is large, leading to a curved decision boundary.

Figure 3.1 shows why the linear kernel was not included among the hyperparameters in the grid search, since the data points do not appear to be linearly separable.

Figure 3.1: Scatterplot of the two input features, colored according to the category “Recession” (orange) and “Expansion” (blue).

Chart, scatter chart

Description automatically generated

One training phase has been carried out, including data from January 1982, up to December 2004, with a validation set ranging from January 2005 to April 2013, after which the first evaluation metrics have been computed, followed by a test set ranging from May 2013 to April 2022. This choice follows a similar reasoning by Puglia and Tucker (2020) but is done to force the algorithm to face one or more recessions both in the training and test sets, so that the output can be clearly interpreted.

The authors prioritize forecast accuracy over precision because in their out of sample forecast horizon, no recession had occurred yet, whereas precision is still reported in this framework as the split between training and test sets has been done to draw attention towards the identification of recessions.

The same paper also provides an initial specification of the neural network utilized for comparison with the forecast given by the support vector machine. Puglia and Tucker (2020) build a relatively shallow neural network using two fully connected hidden layers, with nine and five nodes respectively. The authors do not optimize the network architecture, nor the additional hyperparameters, hence the suggested one is taken as initial specification for the forecast in the essay. The hidden layers are initialized with rectified linear unit activation functions, while the output layer is initialized with a sigmoid activation function, and fine tuning is only performed manually on the decision threshold to maximize validation sample F-beta score, holding 0.5 as a lower bound, which will eventually be identified as the optimal parameter for the baseline model, and increasing it in steps of 0.05 up to 0.75, while the number of epochs and the learning rate necessary to minimize the binary cross entropy loss function, depicted in Figure 3.2, are fixed.

Figure 3.2: Binary cross entropy loss as function of number of epochs in training phase.

Chart

Description automatically generated with medium confidence

The network characteristics are summarized in Table 3.1, where the batch size is also included, taken equal to the training set size, thus configuring an example of batch mode gradient descent, due to the limited dimension of the training sample itself. Due to the small size of the neural network, no dropout layer was included among the hidden ones, although an L2 regularization term was added to the second hidden layer, with a default penalty value of 0.01, contributing to lower the onset of overfitting.

Table 3.1: Vanilla neural network architecture as resulting from validation set fine tuning.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Layer** | **Nodes** | **Activation** | **Alpha** | 0.002 |
| 1 | 9 | ReLU | **Threshold** | 0.5 |
| 2 | 5 | ReLU | **Epochs** | 10000 |
| 3 | 1 | Sigmoid | **Batch size** | 275 |

The models discussed so far are then extended by augmenting the input feature set by means of a pair of macroeconomic variables, and in particular, consistently with Chauvet and Piger (2008), monthly data for the Industrial Production Index, having year 2017 as index base, and monthly data for the Total Nonfarm Payroll Employment, to be used along the 10-year to 3-month yield spread, are considered.

To avoid the look-ahead bias, the macroeconomic variables were shifted by one month, so that, for example, jointly with term spread data for March 2023, IP index information for February 2023, released on March 17, 2023, and Nonfarm Payroll information for February 2023, released on March 10, 2023, could be used to generate predictions.

Feature scaling was performed on the newly included variables, since their ranges differ too widely as opposed to the term spread information: more precisely, a logarithmic transformation, coupled with scaling by a factor of 1000, was used for both macroeconomic features, because both are strictly positive by construction, thus not showing any 0, and the process constraints their scales to be much closer to the ranges characterizing term spreads. This allows to boost the algorithm performance and reduces the influence of outliers.

However, no relevant details with respect to feature scaling were outlined by the considered authors, and for this reason a common practice in the machine learning literature was implemented as explained above, hence it must be remarked that this choice may result in misleading interpretations, especially in the case of the Industrial Production Index, whose index number nature translates ambiguously after the scaling process. A more economically meaningful attempt had been made by calculating discrete monthly growth rates of such index, originating negative as well as positive entries for the analyzed samples, but this produced considerably worse forecasts within all the sets, in addition to widening the range between minimum and maximum values too sharply when compared with the term spread information, which is the reason why such attempt was discarded.

1. Results

The initial specification of the support vector machine yields mediocre results for the validation set. The algorithm partially identifies the only recession occurred in the selected validation set, which coincides with Great Financial Crisis of 2007, but is characterized by a coarse behavior, not extracting the structure in the data appropriately. When applying the model to the test set, the performance varies in the main evaluation metrics, as shown in Figure 4.1.

Figure 4.1: SVM forecast results stemming from the hyperparameters chosen with grid search using the validation set on test set (orange), against actual test sample (blue).

A picture containing text, rectangle, screenshot, line

Description automatically generated

The algorithm now entirely identifies the really occurred recession of 2020, in response to the COVID-19 pandemic, but generates more false positives, and it is interesting to note how the radial basis function kernel is strictly preferred by the grid search, differently from what Puglia and Tucker (2020) conclude, although their feature set is characterized by additional variables, not appearing in this framework.

The out of sample accuracy, precision and recall measures respectively amount to 0.72, 0.32 and 1.0, signaling a low probability of identifying a recession when it really occurs, and even if the recall metric is apparently flawless, this must not be misinterpreted, and its relevance must be put into context, because the test set only contains a relatively short series of true positives, and a long series of true negatives, hence the effort made by the model to recognize periods of expansions when they actually are recessions is not that relevant, at least in the considered test set. Most importantly, when observing the F-beta score, allowing to attribute more relevance to precision, which is important in this framework, due to the imbalance in the dataset, it is trivial to notice that in both cases the values are low.

Figure 4.2 depicts the model out of sample forecast, between April 2022 and March 2023, of high probabilities of incurring in a recession, starting in the following 12 months, albeit a sensible model extension is necessary, and the evaluation metrics do not warrant a reliable prediction.

Figure 4.2: 12-month ahead recession probabilities calculated with term spreads data up to March 2023, as predicted by the support vector machine discussed above.

Chart

Description automatically generated

The neural network provides a somewhat better validation fit, with only the recall rate being slightly worse than the one stemming from the support vector machine and behaves sensibly better compared to the support vector machine on the selected test set, shown in Figure 4.3.

Figure 4.3: NN forecast results stemming from the hyperparameters chosen using the validation set on test set (orange), against actual test sample (blue).

A picture containing chart

Description automatically generated

The seemingly outstanding performance of the neural network must be contextualized: in fact, the test set is more similar to the training set, than to the validation set, in terms of the probability distribution of 0 as opposed to 1, with the exact values collected in Table 4.1.

Table 4.1: Distribution of output variables in selected sets.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Set** | **Count 0** | **Count 1** | **Perc 0** | **Perc 1** | **Total** |
| Training | 225 | 50 | 81.82% | 18.18% | 275 |
| Validation | 70 | 30 | 70.00% | 30.00% | 100 |
| Testing | 94 | 14 | 87.04% | 12.96% | 108 |

However, it must also be recognized that the sets have been selected exactly in the same way for the support vector machine, which yields a worse prediction, hence the model comparison is still plausible.

The vanilla neural network generates a very similar forecast, with the recession indicator becoming positive a month later compared to the support vector machine one, as depicted in Figure 4.4.

Figure 4.4: 12-month ahead recession probabilities calculated with term spreads data up to March 2023, as predicted by the neural network discussed above.

Chart, line chart

Description automatically generated

Extending the models by including the aforementioned macroeconomic variables slightly improves the test set performance for both algorithms, although more sensibly for the support vector machine, but actually lowers the validation metrics at the same time, perhaps widening the gap caused by the previously explained imbalance in the distribution of 0 and 1 in the different samples. The evaluation metrics are outlined in Table 4.2, and pointing out at the reduced performance gap between training and validation test reasonably enables to argue that no sensible overfitting has occurred, with the sole exception of the extended neural network model, for which the margin is indeed somewhat wider, but still acceptable.

Table 4.2: Evaluation metrics for the baseline and extended models.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Vanilla SVM** | **Macro SVM** | **Vanilla NN** | **Macro NN** |
| **Train Acc** | 0.75 | 0.74 | 0.85 | 0.85 |
| **Val Acc** | 0.66 | 0.66 | 0.74 | 0.71 |
| **Test Acc** | 0.72 | 0.83 | 0.96 | 0.94 |
| **Train Prec** | 0.44 | 0.41 | 0.65 | 0.81 |
| **Val Prec** | 0.44 | 0.44 | 0.60 | 0.54 |
| **Test Prec** | 0.32 | 0.44 | 0.81 | 1.00 |
| **Train Rec** | 0.64 | 0.55 | 0.44 | 0.26 |
| **Val Rec** | 0.50 | 0.47 | 0.40 | 0.23 |
| **Test Rec** | 1.00 | 1.00 | 0.93 | 0.50 |
| **Train F-beta** | 0.47 | 0.43 | 0.59 | 0.57 |
| **Val F-beta** | 0.45 | 0.44 | 0.55 | 0.43 |
| **Test F-beta** | 0.37 | 0.49 | 0.83 | 0.83 |

Another interesting remark is that, when extending the specification with macroeconomic information, the grid search identifies a polynomial kernel of degree 2 as the optimal choice for improving the prediction, more in line with the results by Puglia and Tucker (2020).

The support vector machine outputs a forecast on the test set as depicted in Figure 4.5, which more accurately identifies the COVID-19 related recession and yields a lower number of false positives farther away, which distinguished the baseline forecast shown in Figure 4.1. The 12-month ahead prediction is identical to the one generated by the vanilla support vector machine, and for this reason the related figure is not included in the paper but can be replicated using the code in the Appendix.

Figure 4.5: Extended SVM forecast results stemming from the hyperparameters chosen with grid search using the validation set on test set (orange), against actual test sample (blue).

A picture containing text, screenshot, line, rectangle

Description automatically generated

The neural network is influenced in a similar way, with a smaller number of false positives farther away from the only recession occurring in the test sample, but at the same time skips a few true positives which were correctly predicted by the vanilla counterpart, hence resulting in an equivalent F-beta score, as depicted in Figure 4.6.

Figure 4.6: Extended NN forecast results stemming from the hyperparameters chosen using the validation set on test set (orange), against actual test sample (blue).

A picture containing text, screenshot, diagram, rectangle

Description automatically generated

However, the new specification offers a somewhat different 12-month ahead for the out of sample forecast, with the recession indicator firing up later in time compared to the previous ones, as described in Figure 4.7.

Figure 4.7: 12-month ahead recession probabilities calculated with term spreads data up to March 2023, as predicted by the extended neural network discussed above.

A picture containing text, screenshot, line, diagram

Description automatically generated

1. Replication

Engstrom and Sharpe (2018) formulate a probit specification to forecast recession probabilities, using near-term forward yield spreads as predictors, and compare their explanatory power to the one of the most often utilized features, such as the spread between the 10-year Treasury bond and the 2-year counterpart.

The rationale is that such forward spreads offer a more economically representative interpretation, since they measure market expectations with respect to the near-term path of monetary policy, and the authors claim that bond yields beyond 18 months in maturity offer no added value for forecasting purposes.

More precisely, Engstrom and Sharpe (2018) use quarterly data from 1972:Q1 to 2018:Q2 to form the dependent variable time series of recession indicators as provided by NBER, with the modification that the indicator is equal to 1 if a recession occurred at any of the subsequent 4 quarters. It is important to note that observations in which the economy was already in a recession in the previous quarter have been further modified by the authors, allowing the model to strictly capture the probability of transitioning into a recession, along with any observation for which the effective lower bound was binding, because that constrained the near-term forward spread measure to be nonnegative.

Quarterly averages are computed from daily yield data to determine the forward rates and the subsequent term spreads, matching the macroeconomic features’ frequency.

Figure 5.1 depicts the estimated recession probabilities resulting from the second probit specification proposed by the authors, which only takes the near-term forward spread as input, and closely resembles Figure 3, available on page 8 of the original paper.

Figure 5.1: Probit-estimated recession probabilities 4-quarters ahead against baseline recession indicators from 1972:Q1 to 2018:Q2 using near-term forward spread as predictor.

Chart, histogram

Description automatically generated

The same methodology is followed to carry out the model comparison, but the considered period ranges from 1981:Q4 to 2023:Q1 since the related data enable to provide an out of sample estimate of the future recession probabilities, in line with the previous implementations. The results are shown in Figure 5.2 and it is interesting to note how the probabilities sharply increase around 2022:Q2, signaling a high likelihood of recession within the following 4 quarters, consistently with the predictions generated by the support vector machine and the neural network.

Figure 5.2: Probit-estimated recession probabilities 4-quarters ahead against baseline recession indicators from 1981:Q4 to 2023:Q1 using near-term forward spread and long-term yield spread as individual predictors.

Chart, histogram

Description automatically generated

Engstrom and Sharpe (2018) conclude that the near-term forward spread is a more valid predictive tool compared to the commonly chosen yield spreads, however the following replication results seem to be at least partially in contrast with their argument. More precisely, the authors fit a specification using both the near-term forward spread and the long-term yield spread measures, with the latter represented by the 10-year minus 2-year spread, and find that the near-term forward spread has a statistically significant sensible impact on recession probabilities, whereas the long-term yield spread loses its marginal effect and its statistical significance when coupled with the examined one, although in the current replication attempt, perhaps due to the different sample period, this is not true, because both measures are highly impactful and statistically significant at the same time, providing a joint forecast which is sharper than the ones shown above. Figure 5.3 shows the outcome of the joint probit specification.

Figure 5.3: Probit-estimated recession probabilities 4-quarters ahead against baseline recession indicators from 1981:Q4 to 2023:Q1 using near-term forward spread and long-term yield spread as joint predictors.

A picture containing text, line, diagram, parallel

Description automatically generated

Specifically, it is possible to observe that using the features jointly allows to avoid the spikes in probabilities occurred during the period when the zero lower bound was indeed binding, generated by the previous models.

Minor changes in the probability regimes occur for the specifications where the near-term forward spread is coupled with alternative measures of the yield slope, such as the 10-year to 1-year yield spread, or the 10-year to 3-month yield spread, but surprisingly the marginal effect of the near-term forward spread becomes less and less significant, with the majority of the forecasting power captured by the yield spreads themselves, contrary to what Engstrom and Sharpe (2018) conclude. The detailed results are available in Table 5.1, outlined below.

Table 5.1: Marginal effects and P-values for the replicated probit specifications over the period 1981:Q4 to 2023:Q1.

|  |  |  |
| --- | --- | --- |
| **Variables** | **Marginal effects** | **P-values** |
| NTFS | -0.2493 | 0.000 |
| 10Y-2Y | -0.3038 | 0.000 |
| NTFS / 10Y-2Y | -0.0999 / -0.1527 | 0.000 / 0.000 |
| NTFS / 10Y-1Y | -0.0775 / -0.1265 | 0.024 / 0.000 |
| NTFS / 10Y-3M | 0.0484 / -0.1889 | 0.437 / 0.000 |

Due to their redundancy, graphs for the probit models containing information about the 10-year to 1-year and the 10-year to 3-month yield spreads were not included in the essay but can be replicated using the code provided in the Appendix.

1. Conclusions

The paper employs a support vector machine and a neural network, in addition to a more standard probit model according to Engstrom and Sharpe (2018), to provide a one-year ahead forecast of recession probabilities, using monthly data for the first two specifications, and quarterly data for the last one. After restricting the input feature set to only contain information about yield spreads, recognized in the literature as one of the most informative recession predictors, the machine learning models are extended to account for a couple of macroeconomic variables, which however only marginally improve the algorithms’ performance on the testing set, hinting at the fact that the term spreads indeed express the largest predictive power among the considered regressors.

All the considered specifications identify the out of sample recession periods, albeit a sensible amount of false positive stems from the support vector machine forecasts, while the neural network generates a more accurate prediction by means of the main evaluation metrics, with the F-beta score chosen as the most representative for the classification task at hand.

Every algorithm, including data up to March 2023, foresees a high recession probability starting in the second quarter of 2022, and the interpretation to be attributed to such a spike in the indicator is that a recession is to be expected at any month or quarter subsequent to 2023:Q2, because due to what the modified series captures, a positive indicator after a row of negative indicators only meets the required condition 12 months ahead. More precisely, the extended neural network outputs a high recession probability at a 12-month ahead horizon starting November 2022, as pointed out in Figure 4.4, later than the other specifications, which always generate a positive indicator between July and August 2022, and the probit model which generates it in 2022:Q2. A single streak of high recession probability periods is forecasted by all the specifications, with the last indicator being generated in March 2023. In order to maintain consistency in the time horizon considered by the machine learning models and the probit specification stemming from the replication attempt, the forecast does not include data beyond March 2023, since the last model takes in quarterly data, and no data for 2023:Q2 is available at the time the essay is being written, namely June 2023, although monthly data up to May 2023 would be usable.

Comparing the baseline NBER recession indicators from the peak through the trough for the same time window allows to conclude that no recession has been observed up to June 2023, when the analysis ceases, but the period spanned by the algorithms ranges from July 2023 to March 2024, thus at the time of the current analysis no recession indicators have been made available beyond May 2023, leaving the question to be answered by means of future comparison.

Room for further improvement of the machine learning methods implementation exists, in particular with respect to the choice of the macroeconomic features and their subsequent preprocessing stage, for which a more careful and meaningful approach most likely can be found, as conveyed in Section 3, and with respect to the validation approach, which led to selecting training and testing samples characterized by a similar probability distribution, most likely responsible for positively biasing the out of sample evaluation metrics, albeit motivated by two fundamental properties of the time series at hand, namely the time ordering factor and the reduced time series length characterizing observations at such a low frequency.

Elaborating more on the last point, this is the reason why fine tuning was kept to a bare minimum for the neural network by only altering the decision threshold, since its performance was already starkly boosted when compared to the support vector machine. The reduced sample length also motivates a data augmentation attempt that could be made by upsampling the recession indicator time series, for example, at a daily frequency, under the assumption that if a given period was identified by NBER as a recession period, then all the related days would also be classified as recession days. This would increase the sample size from approximately 500 entries for the period ranging from January 1982 to March 2023, to beyond 10000, most likely enabling the algorithms to work at an increased potential. This reasoning could be justified by the existence of a daily frequency time series for term spread information, whose missing values only amount to 4% of the total size, evenly distributed across the time series, and never appearing more than twice consecutively. For these characteristics, they could be easily adjusted, for example, by taking the average of the closest past and future existing values, without biasing the time series by adding a pattern where there should be none. The prediction would however be likely characterized by a noisy behavior, thus careful assumptions for classifying the respective month as a recession or expansion period would be necessary, for example by assuming that a month could be classified as recessionary if the majority of the related days would be labelled as recessionary, in this way preventing daily forecasts from being ambiguous or simply uninterpretable, given that the original definition identifies recessions as economic downturns lasting multiple months, and not only multiple days.

References

Banz, R., & Breen, W. J. (1986). Sample-Dependent Results Using Accounting and Market Data: Some Evidence. *Journal of Finance*, *41*(4), 779–793.

Bauer, M., & Mertens, T. M., (2018). Economic Forecasts with the Yield Curve. *FRBSF Economic Letter*, *2018-07*.

Choi, J., Ge, D., Kang, K. H., & Sohn, S. (2023). Yield spread selection in predicting recession probabilities. *Journal of Forecasting*.

Chauvet, M., & Piger, J. (2008). A Comparison of the Real-Time Performance of Business Cycle Dating Methods. *Journal of Business & Economic Statistics*, *26*(3), 42-49.

Berge, T. J., & Jordà, Ò. (2011). Evaluating the Classification of Economic Activity into Recessions and Expansions. *American Economic Journal: Macroeconomics*, *3*(2), 246–277.

McCracken, M. W., & Ng, S. (2015). FRED-MD: A Monthly Database for Macroeconomic Research. *Journal of Business & Economic Statistics*, *34*(4), 574–589.

Kiley, M. T. (2023). Recession Signals and Business Cycle Dynamics: Tying the Pieces Together. *Finance and Economics Discussion Series*, *2023*(008), 1–40.

Ajello, A., Benzoni, L., Schwinn, M., Grigoli, F., & Vazquez-Grande, F. (2022). Monetary Policy, Inflation Outlook, and Recession Probabilities. *FEDS Notes*, *2022-07–12*.

Puglia, M., & Tucker, A. (2020). Machine Learning, the Treasury Yield Curve and Recession Forecasting. *Finance and Economics Discussion Series*, *2020*(038).

Engstrom, E., & Sharpe, S. (2018). The Near-Term Forward Yield Spread as a Leading Indicator: A Less Distorted Mirror. *Finance and Economics Discussion Series*, *2018*(055).

Gogas, P., Papadimitriou, T., Matthaiou, M., & Chrysanthidou, E. (2015). Yield Curve and Recession Forecasting in a Machine Learning Framework. *Computational Economics*, *45*(4), 635–645.

Gogas, P., Papadimitriou, T., & Chrysanthidou, E. (2015). Yield Curve Point Triplets in Recession Forecasting. *International Finance*, *18*(2), 207–226.

Chinn, M. D., & Kucko, K. (2015). The Predictive Power of the Yield Curve Across Countries and Time. *International Finance*, *18*(2), 129–156.

Rudebusch, G. D., & Williams, J. W. (2008b). Forecasting Recessions: The Puzzle of the Enduring Power of the Yield Curve. *Journal of Business & Economic Statistics*, *27*(4), 492–503.

Ang, A., Piazzesi, M., & Wei, M. (2006). What does the yield curve tell us about GDP growth? *Journal of Econometrics*, *131*(1–2), 359–403.

Wright, J. L. (2006). The Yield Curve and Predicting Recessions. *Finance and Economics Discussion Series*, *2006*(07), 1–19.

Chauvet, M., & Potter, S. (2005). Forecasting recessions using the yield curve. *Journal of Forecasting*, *24*(2), 77–103.

Bernard, H., & Gerlach, S. (1998). Does the term structure predict recessions? The international evidence. *International Journal of Finance & Economics*, *3*(3), 195–215.

Estrella, A., & Hardouvelis, G. A. (1991). The Term Structure as a Predictor of Real Economic Activity. *Journal of Finance*, *46*(2), 555–576.

Data appendix

Federal Reserve Bank of St. Louis, NBER based Recession Indicators for the United States from the Peak through the Trough [USRECM], retrieved from FRED, Federal Reserve Bank of St. Louis.

Federal Reserve Bank of St. Louis, 10-Year Treasury Constant Maturity Minus 3-Month Treasury Constant Maturity [T10Y3MM], retrieved from FRED, Federal Reserve Bank of St. Louis.

Federal Reserve Bank of St. Louis, 10-Year Treasury Constant Maturity Minus 2-Year Treasury Constant Maturity [T10Y2YM], retrieved from FRED, Federal Reserve Bank of St. Louis.

Near-term Forward Spread as daily difference between expected 3-month Treasury Bill yield 6-quarters ahead and current 3-month Treasury Bill yield. https://www.neartermforwardspread.com

Board of Governors of the Federal Reserve System (US), Market Yield on U.S. Treasury Securities at 10-Year Constant Maturity, Quoted on an Investment Basis [DGS10], retrieved from FRED, Federal Reserve Bank of St. Louis.

Board of Governors of the Federal Reserve System (US), Market Yield on U.S. Treasury Securities at 2-Year Constant Maturity, Quoted on an Investment Basis [DGS2], retrieved from FRED, Federal Reserve Bank of St. Louis.

Board of Governors of the Federal Reserve System (US), Market Yield on U.S. Treasury Securities at 1-Year Constant Maturity, Quoted on an Investment Basis [DGS1], retrieved from FRED, Federal Reserve Bank of St. Louis.

Board of Governors of the Federal Reserve System (US), Market Yield on U.S. Treasury Securities at 3-Month Constant Maturity, Quoted on an Investment Basis [DGS3MO], retrieved from FRED, Federal Reserve Bank of St. Louis.

U.S. Bureau of Labor Statistics, All Employees, Total Nonfarm [PAYEMS], retrieved from FRED-MD, Federal Reserve Bank of St. Louis-Monthly Database.

Board of Governors of the Federal Reserve System (US), Industrial Production: Total Index [INDPRO], retrieved from FRED-MD, Federal Reserve Bank of St. Louis-Monthly Database.

Appendix. Python code

*### PREPARE DATAFRAME  
# import necessary libraries*import numpy as np  
import pandas as pd  
*# read spread data*short\_spread\_df = pd.read\_excel(io="10y\_3m\_m.xlsx")  
long\_spread\_df = pd.read\_excel(io="10y\_2y\_m.xlsx")  
*# read macro data*ip\_df = pd.read\_excel(io="ip\_m.xlsx")  
nonfarm\_df = pd.read\_excel(io="nonfarm\_m.xlsx")  
*# read recession data*rec\_df = pd.read\_excel(io="rec\_dummy\_m12.xlsx")  
*# convert timestamps to dates*short\_spread\_df["date"] = pd.to\_datetime(short\_spread\_df["date"]).dt.date  
long\_spread\_df["date"] = pd.to\_datetime(long\_spread\_df["date"]).dt.date  
ip\_df["date"] = pd.to\_datetime(ip\_df["date"]).dt.date  
nonfarm\_df["date"] = pd.to\_datetime(nonfarm\_df["date"]).dt.date  
rec\_df["date"] = pd.to\_datetime(rec\_df["date"]).dt.date  
*# convert values to decimals*short\_spread\_df["spread"] = short\_spread\_df["spread"] / 100  
long\_spread\_df["spread"] = long\_spread\_df["spread"] / 100  
*# scale macro data*ip\_df["index"] = np.log(ip\_df["index"]) / 1000  
nonfarm\_df["nonfarm"] = np.log(nonfarm\_df["nonfarm"]) / 1000  
*# merge rate dataframes*merged\_df = short\_spread\_df.merge(right=long\_spread\_df, how="inner", on="date")  
merged\_df = merged\_df.merge(right=ip\_df, how="inner", on="date")  
merged\_df = merged\_df.merge(right=nonfarm\_df, how="inner", on="date")  
merged\_df = merged\_df.merge(right=rec\_df, how="inner", on="date")

*### SUPPORT VECTOR MACHINE  
# import necessary libraries*import matplotlib  
matplotlib.use("Qt5Agg", force = True)  
import matplotlib.pyplot as plt  
import seaborn as sns  
from sklearn import svm  
from sklearn.model\_selection import GridSearchCV, PredefinedSplit  
from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, fbeta\_score  
*# silence UndefinedMetricWarning when running first training due to precision scoring*from sklearn.exceptions import UndefinedMetricWarning  
def warn(\*args, \*\*kwargs):  
 pass  
import warnings  
warnings.warn = warn  
*# prepare merged dataframe*exec(open('prep\_df\_7744044.py').read())  
*# draw scatterplot of spreads colored according to recession indicator*rec\_hue = [0, 1]  
sns.relplot(data = merged\_df, x = 'spread\_x', y = 'spread\_y', hue = 'rec', hue\_order = rec\_hue, aspect = 1.61)  
plt.text(-.01, .02, f"Recession:\t {(merged\_df['rec'][merged\_df['rec'] == 1]).count()}\n".expandtabs(1) +  
 f"Expansion:\t {(merged\_df['rec'][merged\_df['rec'] == 0]).count()}".expandtabs(1))  
*# add title*plt.title("Term spread scatterplot by category")  
*# add y axis title*plt.ylabel("10-year to 2-year spread")  
*# add x axis title*plt.xlabel("10-year to 3-month spread")  
*# create regressor list to set up for loop for baseline and extended model implementation*regr\_lst = [merged\_df.drop(columns = ["date", "index", "nonfarm", "rec"]), merged\_df.drop(columns = ["date", "spread\_y", "rec"])]  
*# split dataset into training and validation sets for prediction*for regr\_arr in regr\_lst:  
 X\_train = regr\_arr[:375].values  
 y\_train = merged\_df["rec"][:375].values  
 X\_val = regr\_arr[275:375].values  
 y\_val = merged\_df["rec"][275:375].values  
 *# set random seed for replicating results* np.random.seed(17)  
 *# set up support vector classifier* svc = svm.SVC(kernel="rbf", degree=1, C=1, gamma=.0005, class\_weight="balanced", random\_state=17)  
 *# set parameter grid for grid search* grid\_params = {"kernel": ["poly", "rbf"], "degree": [2, 3, 4], "C": np.random.randint(low=1, high=500, size=20),  
 "gamma": np.random.uniform(low=.0005, high=500, size=20)}  
 *# create the custom cross validation parameter for gridsearchCV to preserve the time ordering* ps = PredefinedSplit(test\_fold=np.concatenate(([-1] \* 275, [0] \* 100), axis=0))  
 *# run grid search, causes warnings due to no predictions for several combinations, hence no precision score defined* gs = GridSearchCV(estimator=svc, param\_grid=grid\_params, cv=ps, scoring="precision")  
 *# fit classifier to train sets* gs.fit(X\_train, y\_train)  
 *# forecast future recessions* y\_fit = gs.predict(X\_val)  
 *# save best parameters* best\_params\_val = gs.best\_params\_  
 *# save evaluation metrics for validation set* acc\_val = accuracy\_score(y\_true=y\_val, y\_pred=y\_fit, normalize=True)  
 prec\_val = precision\_score(y\_true=y\_val, y\_pred=y\_fit)  
 rec\_val = recall\_score(y\_true=y\_val, y\_pred=y\_fit)  
 fb\_val = fbeta\_score(y\_true=y\_val, y\_pred=y\_fit, beta=.5)  
 *# print evaluation metrics for validation set* print(f"Validation metrics:\t{acc\_val, prec\_val, rec\_val, fb\_val}")  
 *# print best parameters from fine tuning* print(f"Hyperparameters:\t{best\_params\_val}")  
 *# predict on training data for overfitting check with fine tuned hyperparameters* svc = svm.SVC(kernel=best\_params\_val["kernel"], degree=best\_params\_val["degree"], C=best\_params\_val["C"], gamma=best\_params\_val["gamma"], class\_weight="balanced", random\_state=17)  
 *# fit to previous dataset* svc.fit(X\_train, y\_train)  
 *# forecast future recessions* y\_fit = svc.predict(X\_train)  
 *# save evaluation metrics for train set* acc\_tr = accuracy\_score(y\_true=y\_train, y\_pred=y\_fit, normalize=True)  
 prec\_tr = precision\_score(y\_true=y\_train, y\_pred=y\_fit)  
 rec\_tr = recall\_score(y\_true=y\_train, y\_pred=y\_fit)  
 fb\_tr = fbeta\_score(y\_true=y\_train, y\_pred=y\_fit, beta=.5)  
 *# print evaluation metrics for train set* print(f"Training metrics:\t{acc\_tr, prec\_tr, rec\_tr, fb\_tr}")  
 *# specify test set for out of sample evaluation* X\_test = regr\_arr[375:483].values  
 y\_test = merged\_df["rec"][375:483].values  
 *# forecast future recessions* y\_fit = svc.predict(X\_test)  
 *# save evaluation metrics for test set* acc\_oos = accuracy\_score(y\_true=y\_test, y\_pred=y\_fit, normalize=True)  
 prec\_oos = precision\_score(y\_true=y\_test, y\_pred=y\_fit)  
 rec\_oos = recall\_score(y\_true=y\_test, y\_pred=y\_fit)  
 fb\_oos = fbeta\_score(y\_true=y\_test, y\_pred=y\_fit, beta=.5)  
 *# compare forecast with actual series* compare = pd.DataFrame()  
 compare["date"] = merged\_df["date"][375:483]  
 compare["test"] = y\_test  
 compare["fit"] = y\_fit  
 fig, ax = plt.subplots(figsize=(10, 6))  
 compare.plot(x="date", y=["test", "fit"], kind="line", ax=ax).legend(loc="center left")  
 *# add text box to show hyperparameters* props = dict(boxstyle="round", facecolor="white", alpha=.25)  
 ax.text(x=compare["date"][390], y=.8, s=f"Hyperparameters\n" +  
 f"C:\t {best\_params\_val['C']}\n".expandtabs(13) +  
 f"Gamma:\t {round(best\_params\_val['gamma'], 2)}\n".expandtabs(1) +  
 f"Kernel: \t {best\_params\_val['kernel']}\n".expandtabs(2) +  
 f"Degree: \t {best\_params\_val['degree']}".expandtabs(2),  
 verticalalignment='top', bbox=props)  
 *# add text box to show evaluation metrics* ax.text(x=compare["date"][390], y=.4, s=f"Metrics\n" +  
 f"Accuracy:\t {round(acc\_oos, 2)}\n".expandtabs(13) +  
 f"Precision:\t {round(prec\_oos, 2)}\n".expandtabs(5) +  
 f"Recall: \t {round(rec\_oos, 2)}\n".expandtabs(8) +  
 f"Fb score: \t {round(fb\_oos, 2)}".expandtabs(7),  
 verticalalignment='top', bbox=props)  
 *# add title* plt.title("Recession probability forecast from 2013 to 2022")  
 *# add y axis title* plt.ylabel("Probability threshold\n(Low / High)")  
 *# add x axis title* plt.xlabel("Time")  
 *# create array for forecast next 12 month dummies* future\_X = regr\_arr[483:].values  
 *# generate forecast* y\_future = svc.predict(future\_X)  
 *# generate forecast dataframe* future\_df = pd.DataFrame()  
 future\_df["date"] = pd.date\_range(start='2022-04-01', freq="MS", periods=12).date  
 future\_df["rec"] = y\_future  
 *# plot graph for out of sample prediction* fig, ax = plt.subplots(figsize=(10, 6))  
 future\_df.plot(x="date", y="rec", kind="line", ax=ax)  
 *# add title* plt.title("12-month ahead recession probability forecast from 2022 to 2023")  
 *# add y axis title* plt.ylabel("Probability threshold\n(Low / High)")  
 *# add x axis title* plt.xlabel("Time")

*### NEURAL NETWORK TRAINING  
# import necessary libraries*import matplotlib  
matplotlib.use("Qt5Agg", force = True)  
import matplotlib.pyplot as plt  
import tensorflow as tf  
from keras import regularizers  
*# prepare merged dataframe*exec(open('prep\_df\_7744044.py').read())  
*# create regressor dictionary for model implementation*regr\_dct = {"vanilla\_net": merged\_df.drop(columns=["date", "index", "nonfarm", "rec"]),  
 "macro\_net": merged\_df.drop(columns=["date", "spread\_y", "rec"])}  
for k, v in regr\_dct.items():  
 *# split dataset into training set for network training* X\_train = v[:275].values  
 y\_train = merged\_df["rec"][:275].values  
 *# set up neural network for prediction* model = tf.keras.Sequential()  
 model.add(tf.keras.Input(shape=(len(v.columns),)))  
 model.add(tf.keras.layers.Dense(units=9, activation="relu"))  
 model.add(tf.keras.layers.Dense(units=5, activation="relu", kernel\_regularizer=regularizers.l2(.01)))  
 model.add(tf.keras.layers.Dense(units=1, activation="sigmoid"))  
 opt = tf.keras.optimizers.Adam(learning\_rate=.002)  
 model.compile(optimizer=opt, loss='binary\_crossentropy', metrics=[tf.keras.metrics.Precision(thresholds=.5)])  
 *# fit to training set* hist = model.fit(X\_train, y\_train, batch\_size=275, epochs=10000, verbose=0)  
 *# save model to given path* model.save(filepath=k)  
 *# plot the loss as a function of the amount of epochs* loss\_vals = hist.history["loss"]  
 epochs = range(1, len(loss\_vals) + 1)  
 fig, ax = plt.subplots(figsize=(10, 6))  
 plt.plot(epochs, loss\_vals, label="Training loss")  
 plt.title("Binary cross entropy loss as function of epochs")  
 plt.xlabel("Epochs")  
 plt.ylabel("Loss")  
 plt.legend()

*### NEURAL NETWORK REPLICATION  
# import necessary libraries*import matplotlib  
matplotlib.use("Qt5Agg", force = True)  
import matplotlib.pyplot as plt  
import tensorflow as tf  
from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, fbeta\_score  
*# prepare merged dataframe*exec(open('prep\_df\_7744044.py').read())  
*# create regressors dictionary for model implementation, where the second element in the value list is the optimal threshold  
# chosen on the validation set*regr\_dct = {"vanilla\_net\_repl": [merged\_df.drop(columns=["date", "index", "nonfarm", "rec"]), .5],  
 "macro\_net\_repl": [merged\_df.drop(columns=["date", "spread\_y", "rec"]), .55]}  
for k, v in regr\_dct.items():  
 *# split dataset into validation set for hyperparameter tuning* X\_val = v[0][275:375].values  
 y\_val = merged\_df["rec"][275:375].values  
 *# load model from given path* model = tf.keras.models.load\_model(k)  
 *# predict on validation data* y\_fit = model(X\_val, training=False).numpy()  
 *# compare forecast with actual series* compare = pd.DataFrame()  
 compare["date"] = merged\_df["date"][275:375].reset\_index(drop=True)  
 compare["val"] = y\_val  
 compare["fit"] = np.where(np.squeeze(y\_fit) > v[1], 1, 0)  
 *# compute evaluation metrics* acc\_val = accuracy\_score(y\_true=y\_val, y\_pred=compare["fit"].values, normalize=True)  
 prec\_val = precision\_score(y\_true=y\_val, y\_pred=compare["fit"].values)  
 rec\_val = recall\_score(y\_true=y\_val, y\_pred=compare["fit"].values)  
 fb\_val = fbeta\_score(y\_true=y\_val, y\_pred=compare["fit"].values, beta=.5)  
 *# print evaluation metrics for validation set* print(f"Validation metrics:\t{acc\_val, prec\_val, rec\_val, fb\_val}")  
 *# create training set for evaluation metrics for overfitting check* X\_train = v[0][:275].values  
 y\_train = merged\_df["rec"][:275].values  
 y\_fit = model(X\_train, training=False).numpy()  
 *# compare forecast with actual series* compare = pd.DataFrame()  
 compare["date"] = merged\_df["date"][:275].reset\_index(drop=True)  
 compare["train"] = y\_train  
 compare["fit"] = np.where(np.squeeze(y\_fit) > v[1], 1, 0)  
 *# compute evaluation metrics* acc\_tr = accuracy\_score(y\_true=y\_train, y\_pred=compare["fit"].values, normalize=True)  
 prec\_tr = precision\_score(y\_true=y\_train, y\_pred=compare["fit"].values)  
 rec\_tr = recall\_score(y\_true=y\_train, y\_pred=compare["fit"].values)  
 fb\_tr = fbeta\_score(y\_true=y\_train, y\_pred=compare["fit"].values, beta=.5)  
 print(f"Training metrics:\t{acc\_tr, prec\_tr, rec\_tr, fb\_tr}")  
 *# predict on test data* X\_test = v[0][375:483].values  
 y\_test = merged\_df["rec"][375:483].values  
 y\_fit = model(X\_test, training=False).numpy()  
 *# compare forecast with actual series* compare = pd.DataFrame()  
 compare["date"] = merged\_df["date"][375:483].reset\_index(drop=True)  
 compare["test"] = y\_test  
 compare["fit"] = np.where(np.squeeze(y\_fit) > v[1], 1, 0)  
 *# compute evaluation metrics* acc\_oos = accuracy\_score(y\_true=y\_test, y\_pred=compare["fit"].values, normalize=True)  
 prec\_oos = precision\_score(y\_true=y\_test, y\_pred=compare["fit"].values)  
 rec\_oos = recall\_score(y\_true=y\_test, y\_pred=compare["fit"].values)  
 fb\_oos = fbeta\_score(y\_true=y\_test, y\_pred=compare["fit"].values, beta=.5)  
 *# build comparison graph* fig, ax = plt.subplots(figsize=(10, 6))  
 compare.plot(x="date", y=["test", "fit"], kind="line", ax=ax).legend(loc="right")  
 *# add text box to show hyperparameters* props = dict(boxstyle="round", facecolor="white", alpha=.25)  
 ax.text(x=compare["date"][30], y=.8, s=f"Hyperparameters\n" +  
 f"Alpha:\t {0.002}\n".expandtabs(15) +  
 f"Epochs:\t {10000}\n".expandtabs(7) +  
 f"Threshold: \t {v[1]}\n".expandtabs(4),  
 verticalalignment='top', bbox=props)  
 *# add text box to show evaluation metrics* ax.text(x=compare["date"][30], y=.4, s=f"Metrics\n" +  
 f"Accuracy:\t {round(acc\_oos, 2)}\n".expandtabs(13) +  
 f"Precision:\t {round(prec\_oos, 2)}\n".expandtabs(5) +  
 f"Recall: \t {round(rec\_oos, 2)}\n".expandtabs(8) +  
 f"Fb score: \t {round(fb\_oos, 2)}".expandtabs(7),  
 verticalalignment='top', bbox=props)  
 *# add title* plt.title("Recession probability forecast from 2013 to 2022")  
 *# add y axis title* plt.ylabel("Probability threshold\n(Low / High)")  
 *# add x axis title* plt.xlabel("Time")  
 *# create array for forecast next 12 month dummies* future\_X = v[0][483:].values  
 *# generate forecast* y\_future = model(future\_X, training=False).numpy()  
 *# generate forecast dataframe* future\_df = pd.DataFrame()  
 future\_df["date"] = pd.date\_range(start='2022-04-01', freq="MS", periods=12).date  
 future\_df["rec"] = np.where(np.squeeze(y\_future) > v[1], 1, 0)  
 *# plot graph for out of sample prediction* fig, ax = plt.subplots(figsize=(10, 6))  
 future\_df.plot(x="date", y="rec", kind="line", ax=ax)  
 *# add title* plt.title("12-month ahead recession probability forecast from 2022 to 2023")  
 *# add y axis title* plt.ylabel("Probability threshold\n(Low / High)")  
 *# add x axis title* plt.xlabel("Time")

*### PAPER REPLICATION  
# import necessary libraries*import pandas as pd  
import statsmodels as sm  
from statsmodels.discrete.discrete\_model import Probit  
import matplotlib  
matplotlib.use("Qt5Agg", force = True)  
import matplotlib.pyplot as plt  
*# read data*ntfs = pd.read\_excel(io = "ntfs\_d.xlsx")  
ntfs\_new = pd.read\_excel(io = "ntfs\_new\_d.xlsx")  
rec\_q = pd.read\_excel(io = "rec\_dummy\_q4.xlsx")  
rec\_q\_new = pd.read\_excel(io = "rec\_dummy\_new\_q4.xlsx")  
long\_spread = pd.read\_excel(io = "10y\_2y\_d.xlsx")  
medium\_spread = pd.read\_excel(io = "10y\_1y\_d.xlsx")  
short\_spread = pd.read\_excel(io = "10y\_3m\_d.xlsx")  
*# read vanilla recession dataframe for graph comparison from 1972 to 2018*rec\_df = pd.read\_excel(io = "rec\_dummy\_q.xlsx")  
*# read vanilla recession dataframe for graph comparison from 1981 to 2023*rec\_df\_new = pd.read\_excel(io = "rec\_dummy\_new\_q.xlsx")  
*# create quarterly dataframe*ntfs = ntfs.set\_index("date").resample("QS").mean()  
ntfs["date"] = ntfs.index  
ntfs = ntfs.reset\_index(drop = True)[["date", "spread"]]  
ntfs\_new = ntfs\_new.set\_index("date").resample("QS").mean()  
ntfs\_new["date"] = ntfs\_new.index  
ntfs\_new = ntfs\_new.reset\_index(drop = True)[["date", "spread"]]  
long\_spread = long\_spread.set\_index("date").resample("QS").mean()  
long\_spread["date"] = long\_spread.index  
long\_spread = long\_spread.reset\_index(drop = True)[["date", "spread"]]  
medium\_spread = medium\_spread.set\_index("date").resample("QS").mean()  
medium\_spread["date"] = medium\_spread.index  
medium\_spread = medium\_spread.reset\_index(drop = True)[["date", "spread"]]  
short\_spread = short\_spread.set\_index("date").resample("QS").mean()  
short\_spread["date"] = short\_spread.index  
short\_spread = short\_spread.reset\_index(drop = True)[["date", "spread"]]  
*# merge dataframes*merged\_df = ntfs.merge(right = rec\_q, how = "inner", on = "date")  
merged\_df\_new = ntfs\_new.merge(right = long\_spread, how = "inner", on = "date")  
merged\_df\_new = merged\_df\_new.merge(right = rec\_q\_new, how = "inner", on = "date")  
merged\_df\_medium = ntfs\_new.merge(right = medium\_spread, how = "inner", on = "date")  
merged\_df\_medium = merged\_df\_medium.merge(right = rec\_q\_new, how = "inner", on = "date")  
merged\_df\_short = ntfs\_new.merge(right = short\_spread, how = "inner", on = "date")  
merged\_df\_short = merged\_df\_short.merge(right = rec\_q\_new, how = "inner", on = "date")  
*# initialize and fit probit model to data for near-term forward spread from 1972 to 2018*probit\_model\_ntfs = sm.discrete.discrete\_model.Probit(endog = merged\_df["rec"].values, exog = merged\_df["spread"].values)  
model\_res\_ntfs = probit\_model\_ntfs.fit()  
*# initialize and fit probit model to data for near-term forward spread from 1981 to 2023*probit\_model\_ntfs\_new = sm.discrete.discrete\_model.Probit(endog = merged\_df\_new["rec"].values, exog = merged\_df\_new["spread\_x"].values)  
model\_res\_ntfs\_new = probit\_model\_ntfs\_new.fit()  
*# initialize and fit probit model to data for long-term yield spread from 1981 to 2023*probit\_model\_long = sm.discrete.discrete\_model.Probit(endog = merged\_df\_new["rec"].values, exog = merged\_df\_new["spread\_y"].values)  
model\_res\_long = probit\_model\_long.fit()  
*# create dictionary to host repeated predictions for model comparison*df\_dct\_pred = {"long": merged\_df\_new, "medium": merged\_df\_medium, "short": merged\_df\_short}  
for k, v in df\_dct\_pred.items():  
 probit\_model = sm.discrete.discrete\_model.Probit(endog = v["rec"].values, exog = v[["spread\_x", "spread\_y"]].values)  
 model\_res = probit\_model.fit()  
 print(model\_res.get\_margeff().summary())  
 df\_dct\_pred[k] = model\_res.predict()  
*# extract marginal effects for the different models outside the loop*print(model\_res\_ntfs.get\_margeff().summary())  
print(model\_res\_ntfs\_new.get\_margeff().summary())  
print(model\_res\_long.get\_margeff().summary())  
*# predict recession probabilities for the models outside the loop*model\_pred\_ntfs = model\_res\_ntfs.predict()  
model\_pred\_ntfs\_new = model\_res\_ntfs\_new.predict()  
model\_pred\_long = model\_res\_long.predict()  
*# define above dictionary values for repeated graphs below*model\_pred\_ntfs\_long = df\_dct\_pred["long"]  
model\_pred\_ntfs\_medium = df\_dct\_pred["medium"]  
model\_pred\_ntfs\_short = df\_dct\_pred["short"]  
*# plot predicted values against recession indicators from 1972 to 2018*pred\_df = pd.DataFrame()  
pred\_df["date"] = merged\_df["date"]  
pred\_df["fit\_ntfs"] = model\_pred\_ntfs  
pred\_df["rec"] = rec\_df["rec"]  
pred\_df.plot(x = "date", y = ["rec", "fit\_ntfs"], kind = "line", figsize = (10, 6))  
plt.xlabel("Time")  
plt.ylabel("Probability")  
plt.title("Estimated recession probabilities")  
*# plot predicted values against recession indicators from 1981 to 2023 for individual regressors*pred\_df\_new = pd.DataFrame()  
pred\_df\_new["date"] = merged\_df\_new["date"]  
pred\_df\_new["fit\_ntfs"] = model\_pred\_ntfs\_new  
pred\_df\_new["fit\_long"] = model\_pred\_long  
pred\_df\_new["rec"] = rec\_df\_new["rec"]  
pred\_df\_new.plot(x = "date", y = ["rec", "fit\_ntfs", "fit\_long"], kind = "line", figsize = (10, 6))  
plt.xlabel("Time")  
plt.ylabel("Probability")  
plt.title("Estimated recession probabilities")  
plt.legend(loc = (.75, .6))  
*# create dataframe dictionary for repeated graphs of joint regressors from 1981 to 2023*df\_dct = {"model\_pred\_ntfs\_long": merged\_df\_new, "model\_pred\_ntfs\_medium": merged\_df\_medium,  
 "model\_pred\_ntfs\_short": merged\_df\_short}  
for k, v in df\_dct.items():  
 pred\_df = pd.DataFrame()  
 pred\_df["date"] = v["date"]  
 pred\_df["fit"] = eval(k)  
 pred\_df["rec"] = rec\_df\_new["rec"]  
 pred\_df.plot(x = "date", y = ["rec", "fit"], kind = "line", figsize = (10, 6))  
 plt.xlabel("Time")  
 plt.ylabel("Probability")  
 plt.title("Estimated recession probabilities")  
 plt.legend(loc = (.75, .6))