Profit Predictor of Companies using LSTM

1st Laura Cervera Sancho
laura.cervera@estudiantat.upc.edu
Barcelona, Catalunya

2nd Joan Font i Casadevall
joan.font.casadevall@estudiantat.upc.edu
Santa Coloma de Farnes, Catalunya

3rd Ivan Quirante González
ivan.quirante@estudiantat.upc.edu
Sant Feliu de Llobregat, Catalunya

***Abstract*—This paper presents a machine learning approach to predict future company profits using historical financial data collected from a web platform. The goal is to evaluate different regression models and assess their effectiveness in forecasting EBITDA, a key profitability metric. The dataset includes financial indicators such as revenue, expenses, and past profits. The methodology covers data acquisition, preprocessing, exploratory analysis, and model evaluation using metrics like Relative Root Mean Squared Error (RRMSE).**

***Index Terms*—profit prediction, machine learning, regression, financial data, forecasting.**

I. Introduction

In today's financial landscape, the ability to anticipate a company's future performance is highly valuable for investors, analysts, and decision-makers. Accurate profit forecasting supports better investment decisions and more effective risk management.

This project focuses on predicting company profitability using historical financial data extracted from a web-based platform. To do this, we evaluate several machine learning models capable of identifying patterns in time series data and forecasting future outcomes.

The main goal is to predict future financial performance based on past company data. Specifically, we target **EBITDA** (Earnings Before Interest, Taxes, Depreciation, and Amortization), a widely used measure of operational profitability. Its relative stability and comparability across companies make it well suited for this kind of predictive modeling.

II. State of the Art

Financial forecasting has traditionally relied on econometric models and expert judgment. In recent years, machine learning methods have become more common, offering better ways to model complex and non-linear patterns from data.

Previous studies have applied machine learning techniques such as linear regression, decision trees, and neural networks to forecast financial metrics like revenue and stock prices [7]. However, few have focused on profit prediction using real company data collected from web platforms, especially across a diverse range of firms. This gap highlights the need for models that can handle heterogeneous profiles and learn from sequential patterns in financial time series.

III. Proposed approach

To predict a company's future EBITDA, we built a pipeline that starts by collecting financial data from the SABI platform using automated web scraping. After cleaning and filtering the data, we created lag features using the previous three years of EBITDA, and used the next year's value as the prediction target.

We trained three models: SVR, Random Forest, and LSTM. Each model was trained separately for each company, using an 80/20 split to keep the time order. The goal was to compare classical regressors with a deep learning model that could better learn temporal patterns.

IV. Experimentation

A. Data Acquisition

To build the dataset for profit prediction, financial data from companies was collected using automated web scraping techniques. The data source was the **SABI (Sistema de Análisis de Balances Ibéricos)** platform, which provides extensive financial and business information about companies operating in Spain and Portugal.

Given the restricted access to the platform, data extraction was performed through a scripted login using institutional credentials. The process involved the use of *Selenium WebDriver*, a Python library that allows programmatic control of a web browser.

To avoid detection as an automated process, several human-like interaction techniques were implemented, including:

- Random mouse movements and scrolling.
- Simulated human typing delays.
- Controlled pauses and randomized user-agent headers.

Once logged in, the script iteratively accessed and downloaded data ranges (e.g., 50 records per iteration) using the platform's data export functionality. Each export was monitored and verified to ensure successful file download.

The script was designed to:

- Clear browser storage and cookies between sessions,
- Automatically handle multiple tabs and modal windows,
- Simulate real user interaction during export configuration,
- Wait for and confirm completion of each file download before proceeding.

The retrieved files were later processed and aggregated into a structured dataset for model training.

The final dataset contains 2,159 unique companies after filtering. This structured and automated data acquisition pipeline ensured a robust and consistent input for subsequent stages of preprocessing and model training.

B. *Data pre-processing*

The Data Management pipeline transforms the raw .xls outputs from the extraction phase into a clean, consolidated dataset ready for analysis. It comprises the following sequential stages:

1) **Initial Extraction and Pivoting**
   - **Script1** reads all `.xls` files in `rawData`, extracts a predefined list of financial and operational fields, and pivots them so that each year becomes a row and each field a column. The result is a per-company `_resum.csv` file with years 1900–2023 uniformly indexed.
   - **Script2** demonstrates the same pivoting workflow for a single CSV, ensuring consistency across files.
2) **File Renaming**
   - **Script3** standardizes filenames by reading the value in cell B1 of each Excel file and renaming the file to that identifier. This ensures all subsequent processing uses a human-readable company code.
3) **Pruning Unwanted Records**
   - **Script4** deletes any CSV whose name indicates "(EN LIQUIDACIÓN)" or "(EXTINGUIDA)", removing companies in liquidation or dissolved status.
   - **Script6** drops any company file lacking data for one or more years between 2000 and 2023, ensuring each entity has at least one valid datapoint per year.
4) **Year-Based Filtering**
   - **Script5** filters each CSV to retain only rows for years $>$1998, discarding older data outside the paper's scope.
5) **Annotating Company Identifiers**
   - **Script7** adds a new `Empresa` column to manage easily the data derived from the filename (stripped of its `.csv` extension), embedding the company identifier within each record for easier joins and aggregation.
6) **Missing-Value Imputation and Normalization**
   - **Script8** loads the filtered CSVs, converts "n.d." and empty strings to `NaN`, parses the `Año` column as a `datetime` index, and applies linear interpolation for columns with $\leq 50\%$ missingness (else filling with zero). After imputation, the index is reset back to a column.
7) **Final Concatenation**
   - **Script9** reads all cleaned files in `processedData2`, concatenates them into a single DataFrame, and writes out `all_data.csv` in `final_data`, producing the master dataset for downstream analysis.

After filtering and cleaning, the resulting dataset comprises 2,159 unique companies with complete yearly financial data spanning from 2000 to 2023. This 24-year period provides a robust temporal basis for training and evaluating time series models.

C. *Data Analysis and Exploration*

With the cleaned dataset in hand, we proceeded to analyze its structure and prepare the features for model training. First, we verified the absence of missing values. Due to the thorough preprocessing steps described earlier, no `NaN` or undefined values were found.

Given the large number of companies in the dataset, we randomly selected a single company for exploratory analysis in order to perform an unbiased and interpretable study.

Following this, we performed a feature elimination process. The goal of this step was to reduce redundancy, minimize multicollinearity, and improve model interpretability and computational efficiency without compromising predictive performance.

Variables with an absolute Pearson correlation coefficient of 0.99 or higher were deemed nearly identical in their information content. Retaining multiple highly correlated variables introduces instability in model coefficients and can lead to overfitting. This is particularly relevant in profit and loss accounts, where many variables are derived by simply aggregating subtotals or taxes.

The following 18 features were removed based on these criteria:

```
["Ingresos de explotación", "EBIT",
"Resultado Explotación", "Resultado
Actividades Ordinarias", "Resultado
ordinarios antes Impuestos",
"Resultado bruto", "Resultado del
Ejercicio", "Resultado financiero",
"Total pasivo y capital propio",
"Total activo", "Otros fondos
propios", "Valor agregado",
"Pasivo fijo", "Otros pasivos
líquidos", "Gastos financieros y
gastos asimilados", "Otros activos
líquidos", "Otros activos fijos",
"Impuestos sobre sociedades"].
```

This dimensionality reduction step not only improved model interpretability but also reduced training time and helped mitigate potential overfitting. The resulting correlation structure is illustrated in Figure 1.

To better understand the temporal behavior of a key financial indicator, EBITDA, we randomly selected five companies and plotted their EBITDA evolution across the available years, as shown in Figure 2.

This visualization highlights several important aspects:

- The presence of year-to-year variability, which suggests the need for models that can capture temporal dependencies.
- Differences in scale and patterns across companies, justifying the need for normalization and company-specific modeling.
- The presence of non-linear trends and potential discontinuities, which may not be well captured by traditional linear models.

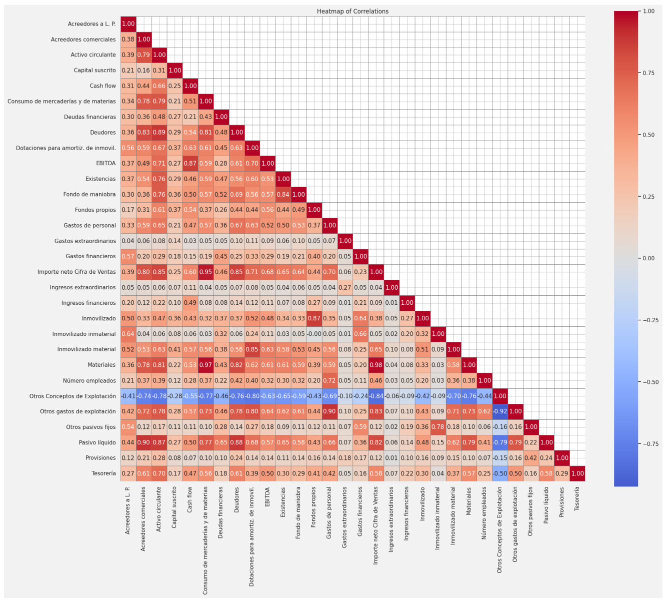

Fig. 1. Triangular correlation heatmap of the remaining financial features across all companies.

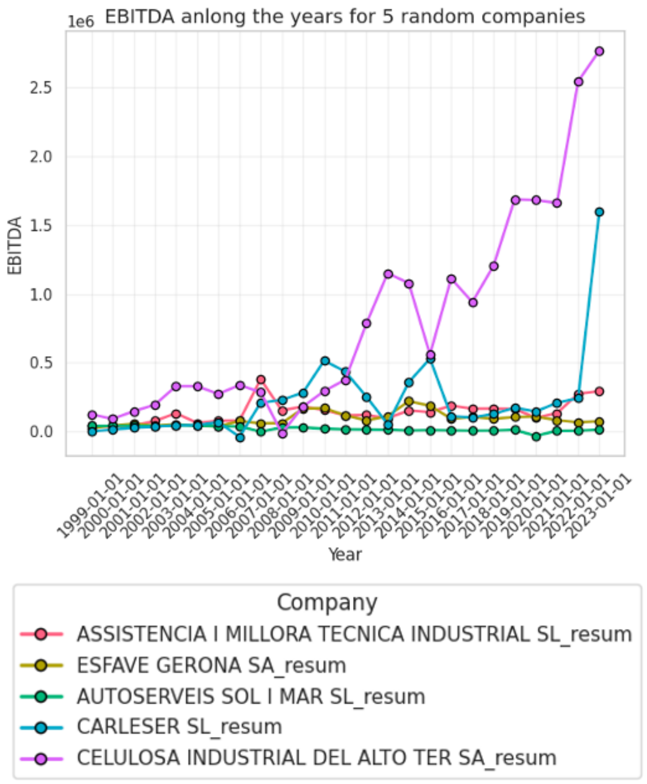

Fig. 2. Evolution of EBITDA over time for 5 randomly selected companies.

As observed in Figure 2, the evolution of EBITDA across five randomly selected companies reveals highly heterogeneous behaviors. While some companies show relatively stable patterns, others present abrupt variations or exponential growth, especially in recent years. Additionally, the magnitude of EBITDA varies drastically between firms, from nearly zero to over 2.5 million euros. This variability in both scale and dynamics makes it extremely difficult to construct a single generalizable model. In fact, during model training, each company was already processed independently: time series were segmented, scaled, and modeled on a per-company basis. Given this setup—and to avoid redundant computation and noisy aggregation of inconsistent behaviors—we opted to focus on a single representative company. This allows for clearer analysis, better interpretability, and more coherent time series modeling, without the confounding effects of inter-company variability.

D. Baseline Models: SVR and RFR

Before introducing deep learning models, we implemented two classical regression algorithms to establish baseline performance: Support Vector Regression (SVR) and Random Forest Regressor (RFR). These models provide interpretable and well-understood benchmarks to assess the relative benefit of more complex architectures such as LSTM.

Support Vector Regression (SVR) is a kernel-based method that seeks to find a function within a specified margin of tolerance ϵ from the true targets. It uses kernel functions—specifically the Radial Basis Function (RBF) in our case—to model non-linear relationships in the input space. SVR is particularly effective for medium-sized datasets and can capture complex patterns without requiring extensive tuning or architecture design.

Random Forest Regressor (RFR) is an ensemble learning method that constructs multiple decision trees during training and outputs their average prediction. It handles non-linearity well, is robust to outliers and noise, and does not require feature scaling. Its ensemble nature also reduces the risk of overfitting that is common in single-tree models.

Both models were trained independently on each company's dataset using recent years' EBITDA values to predict the next one. The data was split chronologically into training (80%) and testing (20%) sets to preserve the temporal structure. After training, each model was evaluated using the Relative Root Mean Squared Error (RRMSE). The results obtained from SVR and RFR serve as baseline scores for comparison with the subsequent LSTM-based deep learning model.

Figures 3 and 4 show the performance of SVR and RFR, respectively, for one representative company. Both models capture the general trend of EBITDA during the training period and extend this pattern into the test period. SVR produces smooth and stable predictions that align closely with the training data. The Random Forest model, which in earlier iterations showed abrupt transitions, now behaves more conservatively, offering smoother test predictions and reduced overfitting.

As shown in Figures 3 and 4, both models follow the historical trend during training but struggle to anticipate changes in the test period. SVR maintains a stable continuation, while RFR captures fluctuations slightly better, though it still con-

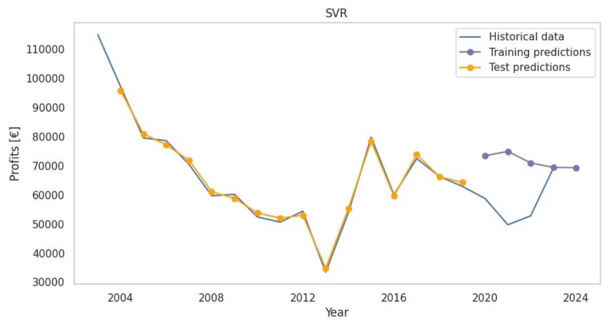

Fig. 3. SVR model predictions for a single company: historical EBITDA, training predictions, and test predictions.

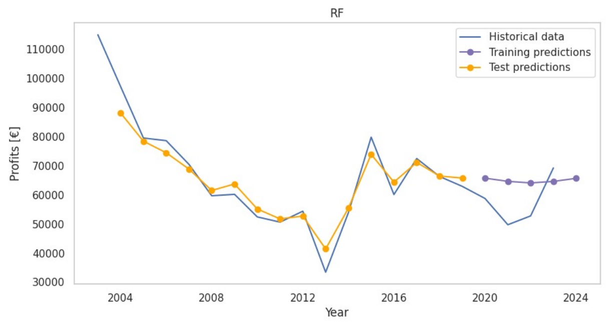

Fig. 4. Random Forest predictions for a single company: historical EBITDA, training predictions, and test predictions.

verges toward flat behavior. In many cases, the best predictions were nearly constant, approximating the mean trend. While this leads to lower RRMSE, it fails to reflect the true dynamics of the data. These results suggest that the limitations lie not only in the models, but also in the data itself, which may lack consistent patterns for accurate forecasting. This highlights the importance of more expressive models like LSTM and the need for richer, more informative datasets.

During the test period, both models show limited ability to anticipate changes observed in the ground truth. SVR maintains a smooth and stable trend, but underreacts to shifts, resulting in conservative forecasts. The Random Forest model follows some of the fluctuations more flexibly, yet still tends to produce flattened predictions. In many cases, the lowest error was achieved by outputting values close to the mean trend, which reduces RRMSE but fails to capture the real dynamics of the series. This reflects both modeling limitations and a lack of strong temporal patterns in the available data. These observations reinforce the need for more expressive models like LSTM, along with richer and more informative datasets.

E. LSTM Model and Training

To address the limitations of classical regression models in capturing temporal dynamics, we implemented a Long Short-Term Memory (LSTM) neural network. LSTMs are a type of Recurrent Neural Network (RNN) specifically designed to handle long-term dependencies in sequential data by incorporating gated mechanisms that control information flow over time. This makes them particularly well-suited for modeling financial time series like EBITDA, which can exhibit both smooth trends and sudden changes.

The input to the LSTM model consisted of fixed-length sequences of past EBITDA values, using a sliding window of three years (i.e., lag 1 to lag 3). Each sample thus represents a time series segment of three years, and the model is trained to predict the EBITDA value of the following year.

The architecture of the network is composed of the following layers:

- `Bidirectional LSTM (64 units)` with `return_sequences=True`
- `Dropout (0.2)`
- `Bidirectional LSTM (32 units)` with `return_sequences=True`
- `Dropout (0.2)`
- `LSTM (16 units)` with `return_sequences=True`
- `Dropout (0.2)`
- `LSTM (8 units)`
- `Dense (16)` $\rightarrow$ `Dense (8)` $\rightarrow$ `Dense (1)`

To avoid overfitting and improve convergence, we used early stopping with a patience of 10 epochs and a learning rate scheduler (`ReduceLROnPlateau`) to reduce the learning rate when the validation loss plateaued. The optimizer used was `Adam`, and the loss function was Mean Squared Error (MSE).

During training, we monitored both RRMSE and MSE on the training and validation sets. The learning curves showed consistent convergence and no significant overfitting, confirming that the model was well-regularized.

Figure 5 shows the prediction performance of the LSTM model for the same company as in the previous baselines. Compared to SVR and RF, the LSTM better captures local variations and adapts more smoothly to trend changes during the test period. Although slight under- and over-predictions still occur, the model shows a stronger ability to generalize temporal patterns.

As shown in Figure 5, the LSTM model successfully captures the historical trend and produces a smoother forecast compared to SVR and RFR. During the training and test periods, it tracks the EBITDA evolution more closely, showing less abrupt jumps and better alignment with the real values. Notably, the forecast beyond the test window (2024–2032) displays a consistent and plausible growth trajectory, in contrast to the flat or noisy extrapolations of the previous models. While the long-term forecast should be interpreted cautiously due to the inherent uncertainty of financial data, the results suggest that the LSTM architecture is better equipped to model sequential dependencies and project future behavior.

F. Evaluation and Results

To evaluate the performance of the three models—SVR, Random Forest Regressor, and LSTM—we used the Relative Root Mean Squared Error (RRMSE) as the main metric. This metric was computed on the test set, capturing the average

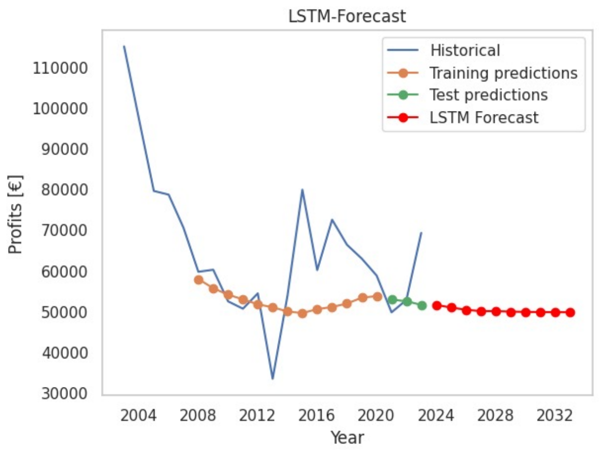

Fig. 5. LSTM model predictions for a single company: historical EBITDA, training predictions, and test predictions.

magnitude of the prediction errors while giving more weight to larger deviations.

Figure 6 shows the RRMSE values for each model applied to the same company. As expected, both classical models provide a reasonable baseline, with SVR slightly outperforming Random Forest. However, the LSTM model significantly reduces the prediction error, demonstrating its superior ability to learn complex temporal dependencies.

These results confirm that, while simpler models can approximate general trends, deep learning approaches like LSTM offer a more precise and stable prediction framework for financial time series. The difference in RRMSE also justifies the added complexity of the LSTM architecture in this context.

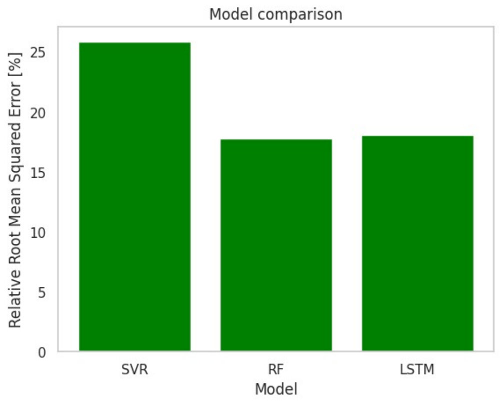

Fig. 6. Relative Root Mean Squared Error (RRMSE) comparison among SVR, RF, and LSTM models.

To conclude the evaluation, we present a global comparison of prediction accuracy across all companies and models. Figure 7 shows a scatter plot of predicted versus real EBITDA values for SVR, Random Forest, and LSTM. Each point corresponds to a forecast, with marker shape indicating the model and color representing the prediction year. Additionally, the Mean Absolute Percentage Error (MAPE) is shown for each model.

This global visualization provides a direct comparison of how well each model generalizes across a diverse set of companies.

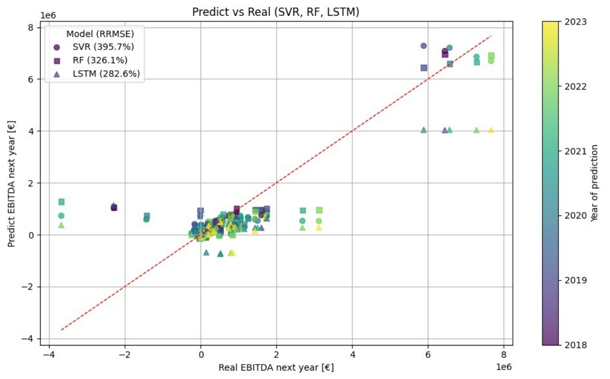

Fig. 7. Predicted vs Real EBITDA values across all companies and models.

Figure 7 reveals considerable dispersion, especially for higher EBITDA values, and none of the models consistently aligns with the identity line. Although LSTM achieves the lowest MAPE, all models report high RRMSE values, reflecting substantial deviation from actual profits. These results suggest that, under the current conditions, the models are not reliable enough for practical use. While this reinforces the need for more expressive architectures, it also highlights the importance of improving the quality and informativeness of the available data.

V. Conclusions

In this study, we explored the problem of financial forecasting using real-world data from Spanish companies, focusing on predicting EBITDA one year ahead. The dataset, extracted via web scraping from the SABI platform, was thoroughly preprocessed to ensure consistency and temporal structure. Due to the high variability and lack of homogeneity across firms, we focused on a single representative company to ensure modeling consistency.

Three modeling approaches were evaluated: Support Vector Regression (SVR), Random Forest Regressor (RFR), and a deep learning model based on Long Short-Term Memory (LSTM) networks. Among them, the LSTM produced better results in terms of RRMSE and alignment with the ground truth, showing better capacity to adapt to temporal dependencies.

However, none of the models achieved accuracy levels suitable for real-world deployment. In many cases, the best predictions were nearly flat, approximating the mean trend rather than capturing real fluctuations. This suggests limitations not only in the models but also in the data, which lacks strong and consistent patterns across time and companies. The evaluation also revealed high RRMSE values, indicating that even the best-performing model still makes large prediction errors in absolute terms.

Future work should focus on improving data quality and exploring more advanced models such as transformers or hybrid architectures, as well as incorporating external information that may enrich the predictive signal.

References

[1] https://colab.research.google.com/github/trekhleb/machine-learning-experiments/blob/master/experiments/text_generation_shakespeare_rnn/text_generation_shakespeare_rnn.ipynb

[2] https://www.kaggle.com/code/kamyarazar/stock-price-prediction-lstm-hyperparameter-tuning#Fearure-Scaling

[3] https://www.baeldung.com/cs/bidirectional-vs-unidirectional-lstm

[4] https://machinelearningmastery.com/visualize-deep-learning-neural-network-model-keras/

[5] https://ieeexplore.ieee.org/document/8663965

[6] https://www.sciencedirect.com/science/article/abs/pii/S0925231212005863

[7] S. Mehtab, J. Sen, and A. Dutta, ”Stock Price Prediction Using Machine Learning and LSTM-Based Deep Learning Models,” arXiv preprint arXiv:2009.10819, 2020.