International College of Economics
and Finance

June 9, 2025

Risk and Return Prediction with LLMs and Financial Texts

Supervisor: Associate Professor, Fabian Slonimczyk

Maxim Shibanov
mvshibanov@edu.hse.ru
https://www.hse.ru/org/persons/1035996664/

Abstract

- I present an evidence that miss-pricing of Fama-French Five factor model could be explained with investors' sentiment.
- I use LLM prompt based methodology developed by Slonimczyk (2024) to extract sentiment scores from 10-K and 10-Q reports of US companies from S&P 500 list for 2018 to 2024.
- I disentangle predictive ability of my sentiment scores by decomposing different types of risk premia from returns.

Fama and French (2015) use dividend discount model to provide theoretical justification for their frame-work:

$$m_t = \sum_{\tau=1}^{\infty} \frac{\mathbb{E}(d_{t+\tau})}{(1+r)^{\tau}}. \tag{1}$$

Where:

- m_t - the share price at time t
- r - the internal rate of return on expected dividends
- $\mathbb{E}(d_{t+\tau})$ - the expected dividend per share for period $t + \tau$

Note, that this model actually mean conditional expectation $\mathbb{E}_{it}(d_{t+\tau})$. And conditional information sets vary time and, more importantly, across investors.

$$m_{it} = \sum_{\tau=1}^{\infty} \frac{\mathbb{E}_{it}(d_{t+\tau})}{(1+r)^{\tau}}. \tag{2}$$

- In frictionless market with no information asymmetry this conditioning makes no difference.
- In real world scenario identity of information sets does not hold.
- Differences in conditional expectations $\rightarrow$ different valuations $\rightarrow$ price impact.
- Hence, there must exist some part of variation in an asset returns that is explained *only* by differences in investors' beliefs.

Motivation
Why Sentiment is needed?

The factors reflect some objective side of a firm business activity which linked to stock returns via assumption of investors rationality and absence of information asymmetry.

↓

Fama-French factors reflect only objective information (implied by firm's actions), that is common across rational investors.

↓

It's natural guess that investors' sentiment might explain price movements that appear due to subjective valuation.

- Fama and French (2015) pointing so called 'lethal' portfolios as one of their main findings. Specifically, these are small companies, that invest a lot relative to their book value.
- In other papers they repeatedly mention that the main problem of the model is small companies.
- Baker and Wurgler (2006) noted that such a companies are exposed to subjective valuation much more than other sorts of firms.

"Bag-of-words" part: The first studies in this field applied a dictionary-based approach, using predefined sets of words with affective modalities.

- Tetlock (2007)
- Loughran and Mcdonald (2011)
- Tetlock (2014)

Achievements:

- Addressing the dimensionality problem of textual data.
- Adequate quantitative representation of text, which had predictive power to stock returns.

"Machine Learning" part:

- Jegadeesh and Wu (2013) - pioneering studies in ML application.
- Bird, Karolyi and Ma (2018) - LDA (Latent Dirichlet Allocation), method for reducing the dimensionality of textual data.
- Ke, B. T. Kelly, and Xiu (2019) - SESTM (Sentiment Extraction via Screening and Topic Modeling), overcome predefined dictionary problem.

"Language models" part:

- Chen, B. T. Kelly, and Xiu (2022) - contextualized embeddings, representing financial news.
- Lopez-Lira and Tang (2024) - theoretical implications of integrating LLMs in investment activity.
- Guo and Hauptmann (2024) - fine-tuning of LLM to predict stock returns.

'bag-of-words' method

- Predefined dictionary with words, grouped affect directions.
- Word-counting for different groups allow to get document level sentiment score.
- Typically sentiment is a count of positive words - count of of negative.
- It is also common to apply different weighting schemes and more complex scalar mapping functions.

$$Polarity = \frac{pos - neg}{pos + neg} \tag{3}$$

Where:

- 'pos' = number of positive words in a document.
- 'neg' = number of negative words in a document.

Mirror Dictionaries

A word could have different meanings, as well as a meaning could have different words. Does a dictionary poses it's predictive power through words or through meanings that stand behind?

↓

To check this, I made mirror dictionaries as follows:

1. Take a word from original dictionary (with it's modality label).
2. Request a list of synonyms from Thesaurus dictionary and exclude all words that existing in original dictionary.
3. Take a random draw from the list with modality label of the original word.

Slonimczyk (2024) method

Following the methodology, of Slonimczyk (2024), I computed sentiment scores using LLM. The main idea of this approach lies in the way of how LLMs are trained:

1. A model receives a text where some word is masked.
2. The model tries to predict this word sampling from a conditional probability distribution of it's vocabulary.
3. Model tweaks it's parameters in a way to maximize the probability of predicting correct word.

This way model "learns" to "understand" and "infer" from natural language.

Slonimczyk (2024) method

Learning procedure gives credit to the following method of sentiment evaluation:

1. Append the prompt with a mask to a piece of text.
2. Ask model to predict this mask.
3. Analyze probability distribution for all vocabulary tokens.
4. Apply a meaningful function to PMF to obtain a scalar value for the piece of text.

Prompt

Choose a prompt that would put LLM in a position of financial advisor. Composing everything onto it's places yields:

"text" + "prompt" + [mask]:

[FINANCIAL REPORT SEGMENT] *Based on this financial report my investment advice is to* **[MASK]**

- The following result is achieved: for each piece of report we obtain the probability distribution for [MASK] token of a model's vocabulary.
- Next step $\rightarrow$ map vector of probabilities into a scalar value.

Verbalizer

- LLM predicts probabilities for its vocabulary (set of tokens).
- Hence, scalar mapping function needs to be defined on tokens' space.

```
{
  "positive": [
    "buy", "invest", "purchase", "Invest", "buying", "stay",
    "proceed", "recommend", "hold", "retain", "increase",
    "maintain", "acquire"
  ],
  "negative": [
    "sell", "avoid", "caut", "carefully", "closely", "caution",
    "analyze", "minimize", "avoid", "decrease", "wait",
    "investigate", "sold", "decline", "monitor", "assess",
    "sale", "remove", "seriously"
  ]
}
```

Scalar mapping function

- Map probabilities of positive and negative tokens to some scalar value.
- Takes tokens from verbalizer as inputs.
- Possibly takes values from -1 to 1 (in fact concentrated around zero).

	good	**excellent**	**up**	**down**	**...**
PMF(vocabulary)	0.0001	0.000203	0.00006	0.082	...

$$\begin{aligned}\text{Score} &= f(PMF(\text{token})) \\ &= P(\text{good}) + P(\text{excellent}) - P(\text{bad}) - P(\text{horrible})\end{aligned}$$

LLM Specification

- Mistral-7B full-precision uninstructed model.
- Context size of 8000 tokens and attention window of 4000 tokens.
- Reports sliced in 8000 token segments (overlap by 20%).

Note, that the model is unable to "read" and "infer" from an overall report. Therefor, each particular sentiment score reflect a mix of a subjective state of mind of a reader with incomplete picture of objective information.

Methodology

Sentiment vectors

1. Each report sliced into 8000 token length pieces.
2. From each slice we get a sentiment score.
3. Different reports have different lengths.
4. Resulting sequences also have different lengths.

I used ”Nearest neighbor interpolation” algorithm to ”stretch” sequences to meet the longest one. This way each entry in a resulting vector corresponds to approximately the same part of report.

Simple returns

$$return_{itf} = \frac{end\_price_{itf} - start\_price_{it}}{start\_price_{it}} * 100\% * \frac{1}{f} \quad (4)$$

Where

- Prefixes 'start' and 'end' mean start and end of time-frame.
- 'i' indicate specific asset.
- 't' - specific trading day.
- 'f' - return time-frame.

Excess returns

$$\text{e\_return}_{itf} = \left(\frac{\text{end\_price}_{itf} - \text{start\_price}_{it}}{\text{start\_price}_{it}} \times 100\% - \frac{\text{s\&p\_end\_price}_{tf} - \text{s\&p\_start\_price}_{t}}{\text{s\&p\_start\_price}_{t}} \times 100\% \right) \times \frac{1}{f} \quad (5)$$

For excess returns I use S&P 500 index returns as a benchmark.

Abnormal returns

Model for returns:

$$return_i = return_f + \beta_m(return_m - return_f) + \beta_s SMB + \beta_h HML + \beta_r RMW + \beta_c CMA + \epsilon_i \quad (6)$$

Where:

- $return_f$ is risk-free return.
- $return_i$ is the return of stock i.
- $return_m$ is the return of the market portfolio.
- SMB (Small Minus Big) captures the size effect.
- HML (High Minus Low) captures the value effect.
- RMW (Robust Minus Weak) captures the profitability effect.
- CMA (Conservative Minus Aggressive) captures the investment effect.

The resulting target variable is a difference between actual return and the one predicted with the model divided by a time-frame length.

Realized volatility

$$\text{r\_vol} = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\left(\log\left(\frac{\text{VWAP}_t}{\text{VWAP}_{t-1}}\right) - \bar{r}\right)^2} \times \sqrt{252 \times 6.5} \tag{7}$$

$$\text{VWAP} = \frac{\sum_{i=1}^{n} P_i \cdot V_i}{\sum_{i=1}^{n} V_i} \tag{8}$$

Where:

- P_i is the price of trade i.
- V_i is the volume of trade i.
- The sum is taken over all trades within the bar's time-frame.
- N is the number of trading hours within particular time-frame. Typically there are 6-7 hours per trading day. So, N=13 for 2 days time-window.
- $\bar{r}$ is the mean of log returns.

Double Fixed-Effects

$$\hat{y}_{itf} = \left(x_{it} - \bar{X}_t - \bar{X}_i\right) \hat{\beta}_{\text{ols}} \tag{9}$$

- Sentiment scores are highly concentrated around zero $\rightarrow$ one unit change might never happen $\rightarrow$ we need to scale.
- Each firm might have it's unique $\sigma_i \rightarrow$ I used within firm scaling.
- Results are robust to different scaling strategy and raw data.

$\hat{\beta}$ here is interpreted as percentage points (for returns) gained per day, given one particular entity standard divination exceedance above average value for this firm across time and plus average value for this time period across entities in the sentiment score.

The Model specification

$$\text{returns}_{itf} = \alpha_i + \lambda_t + \beta_1 \, \text{score}_{itd} + \beta_2 \, \text{firmsize}_{it} + \beta_3 \, \text{epssurprise}_{it} + \beta_4 \, \text{document\_length}_{it} + \beta_5 \, \mathbb{I}\{10\text{Q}\} + e_{it} \quad (10)$$

Where:

- α_i - firm specific effect
- λ_t - time specific affect
- subscript 'd' - the type of sentiment score that are used as a regressor

---

HAC Driscoll-Kraay standard errors are used. Although, the results are robust to other HAC variance estimators. I also checked different bandwidths (lags). Significance don't change from 4 to 12 lags.

Data
Sources

- **Text data:** 10-K and 10-Q reports from the EDGAR database
- **Price data:** Stock prices from the `yfinance` Python library and Alpaca trading API
- **Factor returns:** Kenneth R. French data library

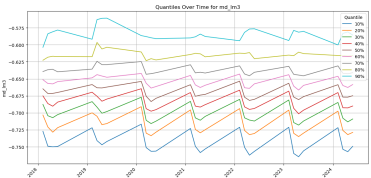
Table: 10-K and 10-Q reports for S&P 500 companies from 2018 to 2024 (inclusive).

Total number of companies processed successfully	498 of 500
Total reports downloaded successfully	13,595
Average document length (characters)	255,469
Average document length (words)	26,908
10-Q reports (count, share)	10,205 (75%)

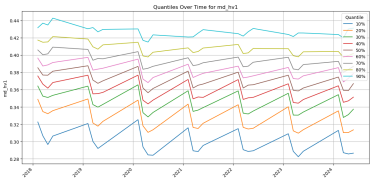
Quintiles of dictionary scores (plotted over time)

(a) LM dict scores

(b) HIV4 dict scores



(c) LM (mirror dict N3) scores



(d) HIV4 (mirror dict N1) scores

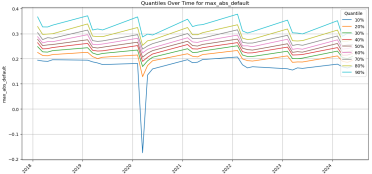
Each document after processing with LLM returns a sequence of a sentiment scores. To use it a scalar predictor we need to summarize it somehow.

I used aggregation functions:

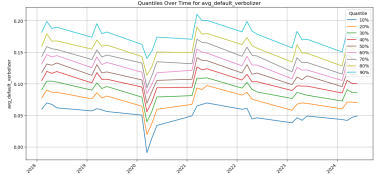
- **Maximum:** Take maximum value from each list.
- **Minimum:** Take minimum value from each list.
- **Average:** Take an average value of from the each list.
- **Maximum Absolute Value:** Take the value that has the maximum absolute value from each list maintaining it's sign.

Since, we have two different verbalizer, overall, we end up with 8 different scores per document.

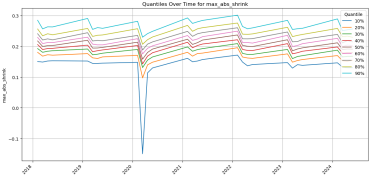
Quintiles of LLM scores (plotted over time)



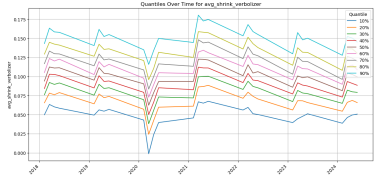
(a) Max abs (default)



(b) Average (default)

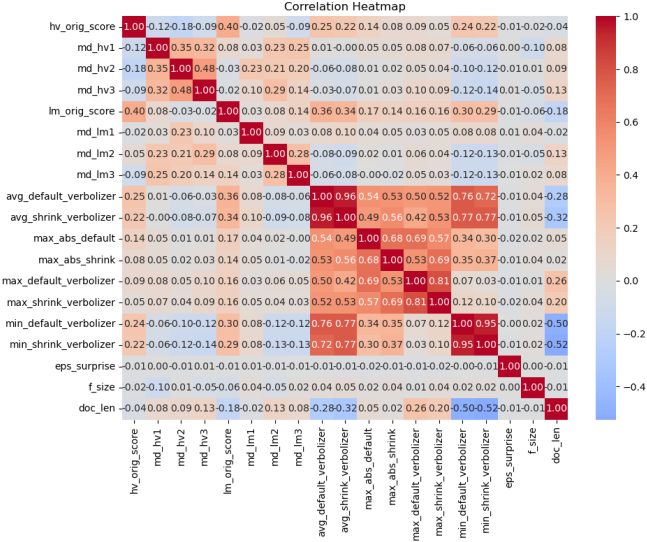


(c) Max abs (shrink)



(d) Average (shrink)

Data
Correlation



Data
Sentiment Vector Distributions

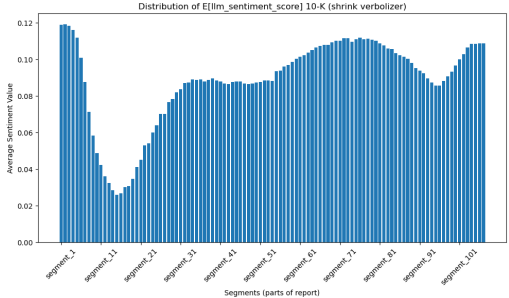


Figure: `10-K reports`

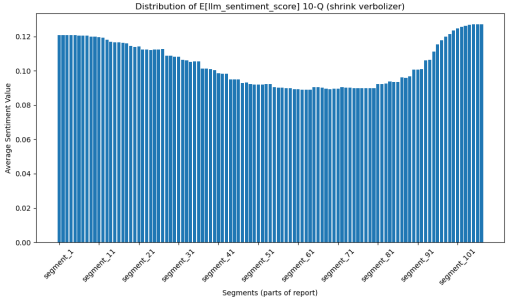


Figure: `10-Q reports`

Results
Dictionary scores

Significance: $p^* < 0.05$, $p^{**} < 0.01$, $p^{***} < 0.001$; SE in parenthesis.

Table: `lm_orig_score`

Target	2d	3d	4d	5d	6d	7d	Q
returns	0.092	0.016	-0.073	0.029	0.009	-0.051	-0.083
	(0.291)	(0.319)	(0.274)	(0.241)	(0.211)	(0.224)	(0.079)
e_returns	-0.028	-0.088	-0.138	-0.005	-0.037	-0.080	-0.075
	(0.248)	(0.275)	(0.244)	(0.210)	(0.180)	(0.169)	(0.064)
abn_returns	0.147	0.045	0.024	0.111	0.086	0.047	-0.023
	(0.167)	(0.184)	(0.163)	(0.143)	(0.135)	(0.127)	(0.043)
r.vol	-0.039	-0.017	-0.030	-0.038	-0.039	-0.036	-0.008*
	(0.025)	(0.019)	(0.024)	(0.021)	(0.021)	(0.018)	(0.004)

Table: `hv_orig_score`

Target	2d	3d	4d	5d	6d	7d	Q
returns	0.441	0.277	-0.063	0.034	0.131	0.124	-0.093
	(0.411)	(0.433)	(0.352)	(0.328)	(0.262)	(0.247)	(0.128)
e_returns	0.478	0.133	-0.068	0.105	0.097	0.085	-0.112
	(0.457)	(0.403)	(0.320)	(0.236)	(0.210)	(0.202)	(0.121)
abn_returns	0.591	0.333	0.171	0.317	0.322	0.315	-0.074
	(0.312)	(0.285)	(0.226)	(0.193)	(0.188)	(0.171)	(0.067)
r.vol	-0.056	0.006	-0.008	-0.014	-0.016	-0.001	-0.032
	(0.031)	(0.046)	(0.036)	(0.033)	(0.033)	(0.036)	(0.027)

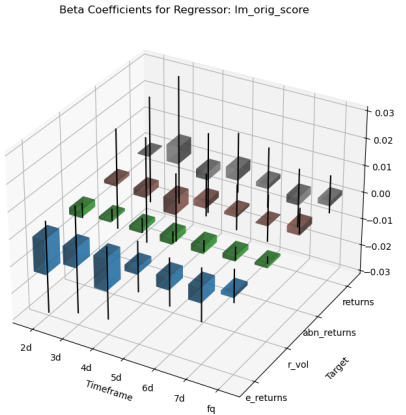
Table: `md_lm3`

Target	2d	3d	4d	5d	6d	7d	Q
returns	1.533***	1.049	0.724	0.526	0.610	0.373	0.038
	(0.522)	(0.710)	(0.624)	(0.570)	(0.431)	(0.466)	(0.051)
e_returns	2.289***	1.687***	1.054*	0.885*	0.900**	0.707*	0.097
	(0.429)	(0.585)	(0.564)	(0.451)	(0.329)	(0.319)	(0.054)
abn_returns	2.025***	1.567***	1.106*	0.943**	0.890**	0.667*	0.196***
	(0.412)	(0.456)	(0.438)	(0.372)	(0.302)	(0.296)	(0.057)
r.vol	-0.111***	-0.122*	-0.125**	-0.134**	-0.121**	-0.122**	-0.018
	(0.033)	(0.050)	(0.045)	(0.049)	(0.044)	(0.044)	(0.037)

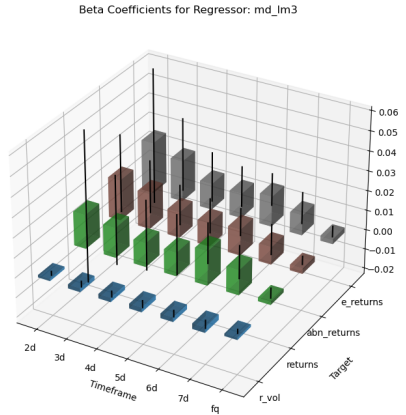
Table: `md_hv1`

Target	2d	3d	4d	5d	6d	7d	Q
returns	1.663	0.630	0.344	0.320	0.381	0.263	0.120
	(0.951)	(0.642)	(0.430)	(0.389)	(0.257)	(0.278)	(0.095)
e_returns	2.096*	1.244	0.742	0.678*	0.693*	0.515**	0.155
	(0.861)	(0.635)	(0.383)	(0.289)	(0.289)	(0.177)	(0.112)
abn_returns	1.121	0.734	0.509	0.546	0.571	0.367*	0.245
	(0.744)	(0.595)	(0.433)	(0.362)	(0.308)	(0.172)	(0.127)
r.vol	-0.048	-0.028	-0.041**	-0.094	-0.065	-0.028	0.004
	(0.027)	(0.029)	(0.017)	(0.050)	(0.048)	(0.049)	(0.044)

Results
Dictionary scores



(a) lm_orig_score



(b) md_lm3

Results
LLM scores

Significance: $p^* < 0.05$, $p^{**} < 0.01$, $p^{***} < 0.001$; SE in parenthesis.

Table: `max_abs_default`

Target	2d	3d	4d	5d	6d	7d	fq
returns	0.0419**	0.0347***	0.0283***	0.0262***	0.0206***	0.0212***	-0.0021
	(0.0144)	(0.0087)	(0.0070)	(0.0054)	(0.0046)	(0.0046)	(0.0018)
e_returns	0.0450**	0.0272**	0.0216**	0.0201**	0.0173**	0.0157**	-0.0014
	(0.0140)	(0.0101)	(0.0073)	(0.0064)	(0.0054)	(0.0051)	(0.0017)
abn_returns	0.0334**	0.0265**	0.0216**	0.0200*	0.0171***	0.0155**	0.0012
	(0.0113)	(0.0093)	(0.0072)	(0.0065)	(0.0052)	(0.0047)	(0.0016)
r.vol	0.0005	-0.0009	-0.0010	-0.0008	-0.0009	-0.0009	-0.0011
	(0.0008)	(0.0011)	(0.0009)	(0.0009)	(0.0009)	(0.0009)	(0.0008)

Table: `avg_default_verbolizer`

Target	2d	3d	4d	5d	6d	7d	fq
returns	0.0242	0.0208	0.0216	0.0207*	0.0181	0.0109	-0.0006
	(0.0200)	(0.0176)	(0.0123)	(0.0099)	(0.0101)	(0.0106)	(0.0033)
e_returns	0.0310	0.0185	0.0176	0.0202*	0.0172	0.0113	-0.0001
	(0.0176)	(0.0160)	(0.0114)	(0.0096)	(0.0089)	(0.0104)	(0.0028)
abn_returns	0.0268**	0.0179*	0.0183**	0.0188***	0.0207**	0.0152*	0.0054***
	(0.0093)	(0.0090)	(0.0065)	(0.0063)	(0.0060)	(0.0072)	(0.0013)
r.vol	-0.0020	-0.0026	-0.0028	-0.0030*	-0.0031*	-0.0028*	-0.0021**
	(0.0012)	(0.0014)	(0.0015)	(0.0013)	(0.0013)	(0.0013)	(0.0008)

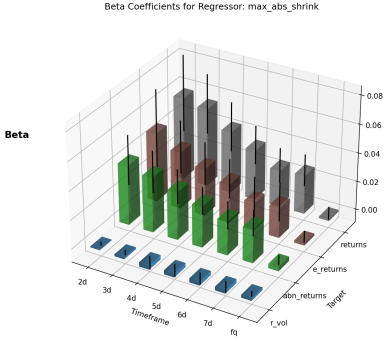
Table: `max_abs_shrink`

Target	2d	3d	4d	5d	6d	7d	fq
returns	0.0576***	0.0545***	0.0429***	0.0348***	0.0253***	0.0267***	0.0012
	(0.0136)	(0.0105)	(0.0087)	(0.0068)	(0.0061)	(0.0055)	(0.0022)
e_returns	0.0478***	0.0395***	0.0327***	0.0269***	0.0193**	0.0209**	0.0015
	(0.0137)	(0.0093)	(0.0075)	(0.0068)	(0.0071)	(0.0068)	(0.0020)
abn_returns	0.0414***	0.0381***	0.0329***	0.0282***	0.0234***	0.0234***	0.0030
	(0.0087)	(0.0073)	(0.0059)	(0.0047)	(0.0047)	(0.0044)	(0.0016)
r.vol	-0.0005	-0.0013	-0.0031	-0.0029	-0.0029	-0.0027	-0.0019*
	(0.0006)	(0.0008)	(0.0022)	(0.0021)	(0.0020)	(0.0020)	(0.0009)

Table: `avg_shrink_verbolizer`

Target	2d	3d	4d	5d	6d	7d	fq
returns	0.0279	0.0283	0.0257	0.0221	0.0199	0.0125	0.0006
	(0.0231)	(0.0200)	(0.0158)	(0.0128)	(0.0122)	(0.0127)	(0.0040)
e_returns	0.0289	0.0198	0.0173	0.0188	0.0169	0.0113	0.0013
	(0.0204)	(0.0184)	(0.0145)	(0.0120)	(0.0106)	(0.0103)	(0.0035)
abn_returns	0.0245*	0.0161	0.0164*	0.0193**	0.0194**	0.0146	0.0062***
	(0.0107)	(0.0099)	(0.0075)	(0.0071)	(0.0070)	(0.0077)	(0.0015)
r.vol	-0.0021	-0.0028	-0.0031*	-0.0033*	-0.0035**	-0.0031*	-0.0025**
	(0.0014)	(0.0015)	(0.0016)	(0.0014)	(0.0013)	(0.0014)	(0.0009)

Results
LLM scores



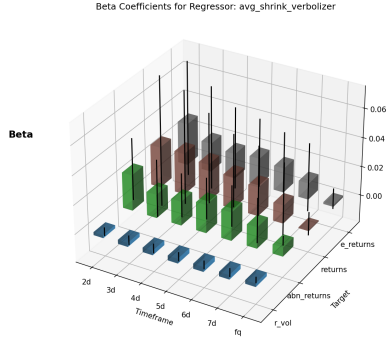
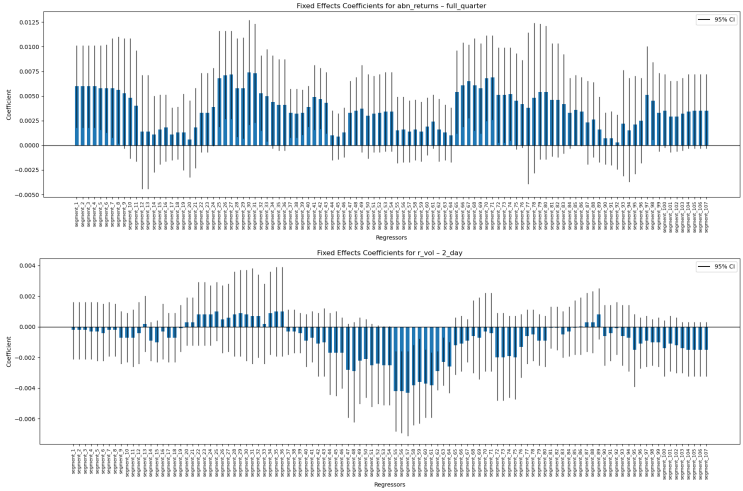


Table: `max_abs_shrink`

Target		2d	3d	4d	5d	6d	7d	fq
returns	Coef.	0.0576***	0.0545***	0.0429***	0.0348***	0.0253**	0.0267***	0.0012
	SE	(0.0136)	(0.0105)	(0.0087)	(0.0068)	(0.0061)	(0.0055)	(0.0022)
	R^2	0.0223	0.0191	0.0183	0.0170	0.0181	0.0189	0.0399
e_returns	Coef.	0.0478***	0.0395***	0.0327***	0.0269***	0.0193**	0.0209**	0.0015
	SE	(0.0137)	(0.0093)	(0.0075)	(0.0068)	(0.0071)	(0.0068)	(0.0020)
	R^2	0.0212	0.0180	0.0170	0.0150	0.0139	0.0135	0.0310
abn_returns	Coef.	0.0414***	0.0381***	0.0329***	0.0282***	0.0234**	0.0234***	0.0030
	SE	(0.0087)	(0.0073)	(0.0059)	(0.0047)	(0.0047)	(0.0044)	(0.0016)
	R^2	0.0238	0.0218	0.0206	0.0189	0.0169	0.0154	0.0326
r.vol	Coef.	-0.0005	-0.0013	-0.0031	-0.0029	-0.0029	-0.0027	-0.0019*
	SE	(0.0006)	(0.0008)	(0.0022)	(0.0021)	(0.0020)	(0.0019)	(0.0009)
	R^2	0.0123	0.0094	0.0090	0.0063	0.0060	0.0061	0.0166

Note: Standard errors in parentheses. *, **, *** indicate significance at 5%, 1%, 0.1%.

Results
Sentiment Vectors



I use Adaptive LASSO on the entire vector to perform feature selection.
Under consistency of initial estimator Adaptive Lasso achieves oracle properties:

- **Selection consistency property:** Probability that the adaptive Lasso identifies the correct set of nonzero coefficients tends to 1 as $n \to \infty$
- **Oracle efficiency:** Asymptotic distribution of the estimated nonzero coefficients is the same as if we knew in advance which variables are truly nonzero

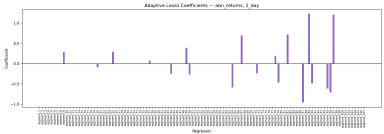
The ALasso $\hat{\beta}$ solves this equation:

$$\hat{\beta}^{\mathsf{AL}} = \arg\min_{\beta} \left\{ \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda_n \sum_{j=1}^{p} w_j |\beta_j| \right\} \tag{11}$$

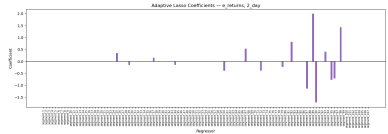
- **Weights:** $w_j = \frac{1}{|\tilde{\beta}_j|^\gamma}$, where $\tilde{\beta}_j$ is a consistent estimator (OLS or Ridge)
- $\gamma > 0$ (commonly used $\gamma = 1$)
- X - matrix composed of sentiment vectors

Results of the test

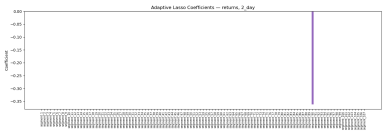
- $w_j = \frac{1}{|\tilde{\beta}_j^{OLS}|^\gamma}$ pushes all coefficients to zero.
- $w_j = \frac{1}{|\tilde{\beta}_j^{RIDGE}|^\gamma}$ yields the following picture:



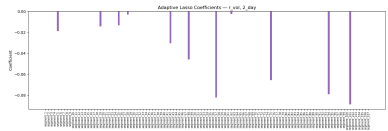
(a) Abnormal Returns



(b) Expected Returns



(c) Raw Returns



(d) Volatility

Robustness Tests
LLM Scores

To account for multiple testing, I employ panel bootstrap ($T = 28$, $N = 498$, $B = 3000$):

1. Apply scaling first (and only once at the beginning).
2. Resample dataset using entire firms as independent draws.
3. Refit the model and store the following statistics:

- $\hat{\beta}_{\text{obs}}$ - Original panel coefficient.
- β_0 - Null hypothesis ($= 0$).
- $\widehat{\text{SE}}_{\text{pan}}$: Panel-robust std. error (e.g., Driscoll–Kraay).
- $\hat{\beta}_b^*$ - Coefficient in bootstrap draw b.
- $\widehat{\text{SE}}_{\text{pan},b}^*$ - s.e. recomputed in draw b.
- Z_{obs} - Observed studentised statistic (on the next slide).
- Z_b^* - Studentised stat in bootstrap draw b(on the next slide).
- B - Bootstrap replications ($B = 3000$).
- $1\{\cdot\}$ - Indicator function.

I use the statistics to compute $\hat{p}_{\text{stud}}$ - One-sided studentised-bootstrap *p*-value: (with the (+1)/(+1) continuity correction)

I computed bootstrap p-values with the following formula:

$$Z_{\text{obs}} = \frac{\hat{\beta}_{obs} - \beta_0}{\widehat{\text{SE}}_{\text{pan}}}, \tag{12}$$

$$Z_b^* = \frac{\hat{\beta}_b^* - \hat{\beta}_{obs}}{\widehat{\text{SE}}_{\text{pan},b}^*}, \qquad b = 1, \ldots, B, \tag{13}$$

$$\hat{p}_{\text{stud}} = \frac{1}{B+1}\left(1 + \sum_{b=1}^{B} 1\{ Z_b^* \geq Z_{\text{obs}} \}\right). \tag{14}$$

Right tail for all types of returns (as we expect $\hat{\beta}_{return} > 0$) and left tail for volatility (as we expect $\hat{\beta}_{vol} < 0$).

How to interpret?

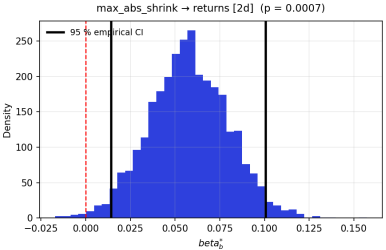
- Effectively, we are asking: **What is the probability to observe that high sampling error $\hat{\beta}_{obs} - \beta_0$ given our data (distribution of resampling error $\hat{\beta}^*_b - \hat{\beta}_{obs}$)?**
- If observed statistic is trivial and we could shuffle our data whatever we want and get similar results $\rightarrow$ **high p-value**.
- If data contain unique dependence and shuffling could erase it $\rightarrow$ **low p-value**.
- **Note**: this test has different meaning from t-test, although they are related.
- **Note**: by construction this test has higher power than t-test.

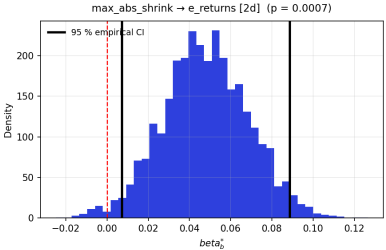
Table: Bootstrap p-vaL and percentile CI for $\hat{\beta}_b^*$ for `max_abs_shrink`

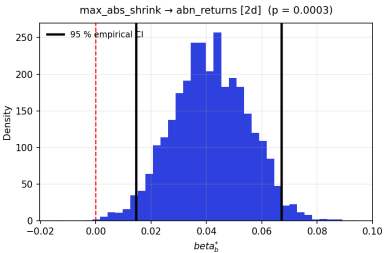
Target		2d	3d	4d	5d	6d	7d	fq
returns	p-val	0.0007***	0.0003**	0.0003***	0.0003***	0.0003***	0.0003***	0.2686
	LB	[0.0142	[0.0251	[0.0182	[0.0155	[0.0082	[0.0103	[-0.0038
	UB	0.1006]	0.0856]	0.0672]	0.0550]	0.0429]	0.0432]	0.0059]
e_returns	p-val	0.0007***	0.0003***	0.0003***	0.0003***	0.0010**	0.0007**	0.2119
	LB	[0.0071	[0.0119	[0.0104	[0.0084	[0.0029	[0.0054	[-0.0031
	UB	0.0887]	0.0680]	0.0550]	0.0453]	0.0361]	0.0365]	0.0060]
abn_returns	p-val	0.0003***	0.0003***	0.0003***	0.0003***	0.0003***	0.0003***	0.0177*
	LB	[0.0146	[0.0171	[0.0147	[0.0126	[0.0096	[0.0099	[-0.0007
	UB	0.0671]	0.0595]	0.0512]	0.0436]	0.0373]	0.0366]	0.0067]
r.vol	p-val	0.1729	0.0080*	0.0003***	0.0003***	0.0003***	0.0003***	0.0003***
	LB	[-0.0022	[-0.0030	[-0.0074	[-0.0068	[-0.0066	[-0.0059	[-0.0036
	UB	0.0012]	0.0003]	-0.0002]	-0.0002]	-0.0005]	-0.0005]	-0.0003]

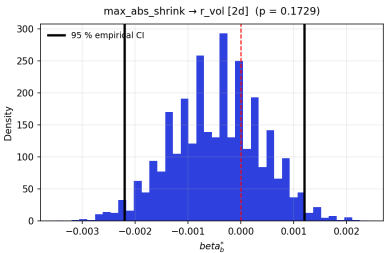
Note: Each block contains p-value (with significance stars), lower bound '[lb' and upper bound 'ub]' of the 95% CI.

Robustness Test
LLM Scores









Discussion
Findings

- Dictionary sentiment scores are not informative about returns.
- LLM sentiment scores from reports have some explanatory power about returns.
- Sentiment scores carry information that is orthogonal Fama-French factors as abnormal returns are more predictable.
- Explanatory power of sentiment scores vanishes after a week (despite long term returns are more predictable in general).
- Maximum absolute value is the best way to summaries sequence of sentiment from report $\rightarrow$ investors are responsive to the highest note in a report (wether it is positive or negative).
- Shrink verbalizer perform better that default one $\rightarrow$ even a relatively small LLM could clearly encapsulate the meaning of financial advise in words "Buy" and "Sell".
- Realized volatility is not predictable with this model specification.
- Not enough statistical evidence that some part of report could carry sentiment scores with superior predictive power.

Conclusion

In this research, using S&P 500 companies' 10-Q and 10-K reports for 2018-2024:

1. Computed dictionary sentiment scores with Loughran and McDonald and Harvard dictionary (as well as with their mirror dictionaries).
2. Computed LLM sentiment scores with Slonimczyk (2024) methodology and tried different aggregation techniques that yields different stat properties.
3. Constructed wide range of targets to decompose different risk-premia and disentangle predictive ability of sentiment.
4. Constructed sentiment vectors and assessed the distribution of sentiment across 10-Q and 10-K report.
5. Went through robustness tests to account for multiple testing problem and see how results are secretive to different SE specifications and scaling strategies.