Big Data in Finance

Week 6: Model Interpretability

Dr Daniele Bianchi

Spring 2026

Today's Agenda

Why Interpretability Matters

Feature Importance Methods

Partial Dependence Plots

SHAP Values

Motivation: Investment Models

Opening the Black-Box

SHAP Analysis

Portfolio Implications

Practical Workflow

Summary

Why Interpretability Matters

The Black Box Problem

We've built powerful models:

- Random Forests with 500 trees
- Boosted trees with hundreds of weak learners
- Models that compete with simple benchmarks

But can we answer these questions?

- *Why* was this loan application denied?
- *What* is driving the model's return forecast?
- *How* would the prediction change if income increased?
- *Which* features matter most for this specific case?

The Challenge

Complex models offer **better predictions** but less **transparency**.
However, in finance, transparency is often required and non-negotiable.

Real-World Consequences: Credit Decisions

Scenario: A loan applicant is denied credit by your ML model.

The applicant asks: "Why was I rejected?"

Without interpretability:

- "The model said no"
- Cannot explain decision
- Potential legal liability
- Customer frustration
- Regulatory non-compliance

With interpretability:

- "Your debt-to-income ratio of 45% was the primary factor"
- Clear, actionable feedback
- Defensible decision
- Regulatory compliance

Legal Requirement

In many jurisdictions, lenders must provide **specific reasons** for adverse credit decisions (e.g., US Equal Credit Opportunity Act).

Real-World Consequences: Investment Decisions

Scenario: Your ML model recommends a large position in a stock.

The portfolio manager asks: "Why does the model like this stock?"

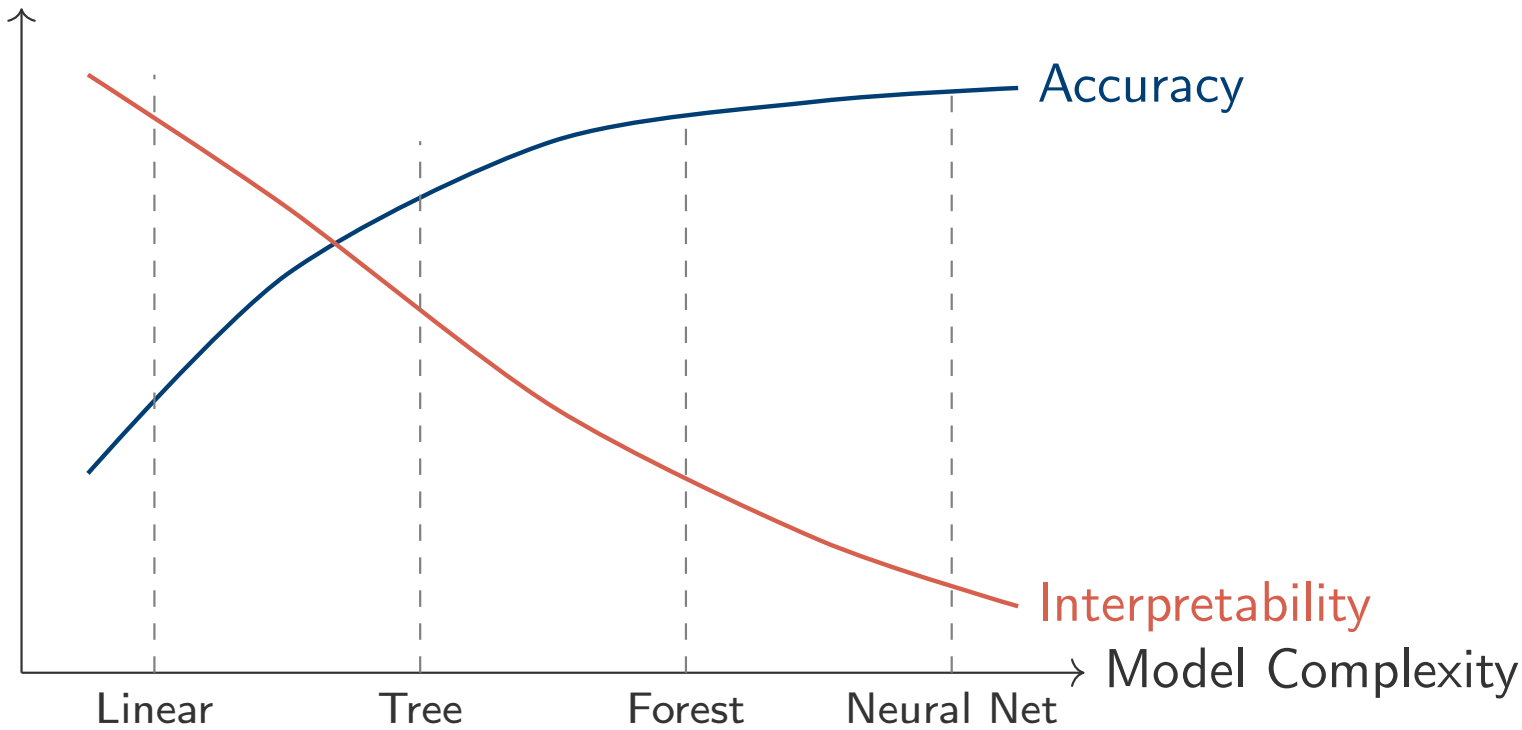
- **Risk management:** Is the model picking up a real signal or noise?
- **Due diligence:** Can we explain this to clients and compliance?
- **Model debugging:** Is the model using sensible features?
- **Regime changes:** Will the signal persist in different conditions?

Example concern: Model heavily weights a feature that is actually a **data artifact** (e.g., ticker symbol encoded as a number).

Trust Through Understanding

Portfolio managers are more likely to **follow** model recommendations they **understand**.

The Accuracy-Interpretability Tradeoff



Traditional view: Must sacrifice accuracy for interpretability.

Modern view: Use **post-hoc interpretability methods** to explain complex models!

Two Approaches to Interpretability

1. **Inherently Interpretable Models:**
   - Linear regression: Coefficients show direction and magnitude
   - Logistic regression: Odds ratios
   - Single decision tree: Visual flowchart
   - Limited complexity $\Rightarrow$ Limited accuracy

2. **Post-Hoc Interpretation of Complex Models:**
   - Keep your Random Forest or XGBoost
   - Apply interpretation methods *after* training
   - Get the best of both worlds

Today's Focus

Post-hoc methods: Explain feature importance

Global vs. Local Interpretability

Two different questions:

Global Interpretability:

"How does the model work *in general*?"

- Which features matter most *overall*?
- What is the average effect of each feature?
- How do features interact?

Methods: Feature importance, PDP

Local Interpretability:

"Why did the model make *this specific* prediction?"

- Why was *this* loan denied?
- What drove *this* stock's forecast?
- How can *this* applicant improve?

Methods: SHAP, LIME

Both Matter

Global: Model validation and debugging. Local: Individual explanations.

Feature Importance Methods

What Is Feature Importance?

Question: Which features contribute most to the model's predictions?

Why we care:

- Identify key drivers of predictions
- Detect potential data leakage (suspicious features)
- Simplify models by removing unimportant features
- Communicate what the model has learned

Two main approaches:

1. **Model-specific:** Built into tree-based models
2. **Model-agnostic:** Permutation importance (works for any model)

Tree-Based Feature Importance

For Random Forest / XGBoost: Built-in importance measure

From Week 3: Regression trees split to reduce variance within nodes

- A good split creates child nodes where outcomes are more similar
- The improvement = variance before split − weighted variance after

Feature importance idea:

- Each split reduces variance by some amount
- Features that create *bigger* variance reductions are more important
- Sum reductions across all splits using that feature, across all trees

$$\text{Importance}(X_j) = \sum_{\text{splits on } X_j} (\text{Variance reduction from split})$$

Advantages: Fast (computed during training).

Disadvantage: Biased toward **high-cardinality** and **correlated** features!

Permutation Feature Importance

Idea: How much does performance *drop* when we break the relationship between a feature and the target?

Algorithm:

1. Train model, compute baseline performance (e.g., R^2, AUC)
2. For each feature X_j:
   - 2.1 Randomly shuffle values of X_j (breaks relationship with Y)
   - 2.2 Compute performance with shuffled X_j
   - 2.3 Importance = Baseline − Shuffled performance
3. Repeat multiple times and average

Intuition

If shuffling X_j hurts performance a lot, then X_j is important.

Permutation Importance: Visualization

Original Data

X_1	X_2	X_3	Y
0.5	12	A	1
0.8	15	B	0
0.3	18	A	1

Shuffle X_2 →

After Shuffling X_2

X_1	X_2	X_3	Y
0.5	**18**	A	1
0.8	**12**	B	0
0.3	**15**	A	1

Baseline MSE: **0.75**

Shuffled MSE: **0.68**

$$\textbf{Importance}(X_2) = \boxed{0.75 - 0.68 = 0.07}$$

Shuffling breaks the relationship between X_2 and Y. The bigger the performance drop, the more important the feature.

Permutation Importance: Advantages and Caveats

Advantages:

- **Model-agnostic:** Works for any model
- Computed on *held-out data* (reflects generalization)
- No cardinality bias
- Intuitive interpretation

Caveats:

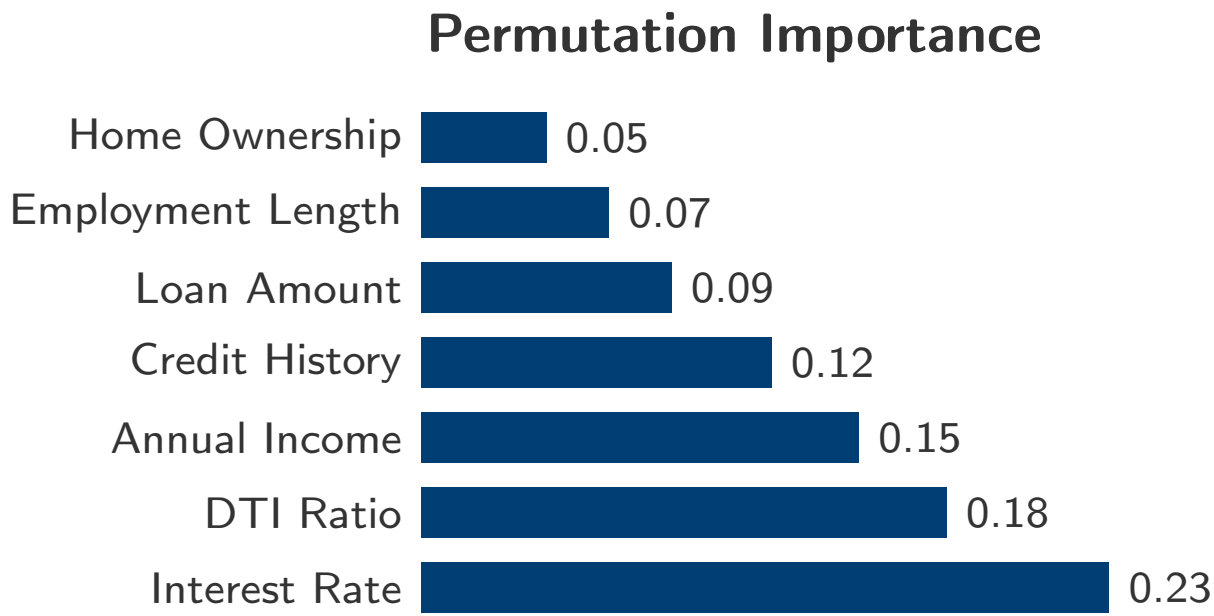
- **Correlated features:** Still problematic!
  - Shuffling one leaves the other intact
  - Model can still predict using the correlated feature
  - Both features appear less important
- **Computational cost:** Need to re-evaluate model many times
- **Extrapolation:** Shuffling can create unrealistic combinations

Rule of Thumb

Use permutation importance, but be cautious with correlated features.

Feature Importance: Credit Risk Example

XGBoost model for loan default prediction:



Interpretation: Interest rate and debt-to-income (DTI) ratio are the strongest predictors of default. This makes economic sense!

Partial Dependence Plots

Beyond Importance: Understanding Relationships

Feature importance tells us *which* features matter.

But we also want to know:

- *How* does the prediction change as a feature changes?
- Is the relationship linear? Monotonic? Non-monotonic?
- Are there threshold effects or interactions?

Partial Dependence Plots (PDP): Show the **marginal effect** of a feature on predictions.

Question PDP Answers

"On average, how does the predicted probability of default change as debt-to-income ratio increases from 10% to 50%?"

Partial Dependence: The Idea

Goal: Isolate the effect of feature X_j on predictions

Definition:

$$\text{PD}(x_j) = \frac{1}{n} \sum_{i=1}^{n} \widehat{f}(x_j, X_{-j}^{(i)})$$

In words:

1. Pick a value x_j for the feature of interest
2. For *every* observation in the data, set $X_j = x_j$ but keep other features unchanged
3. Compute predictions for all these modified observations
4. Average the predictions

Repeat for a grid of values of x_j and plot.

Partial Dependence: Step-by-Step Example

Goal: Isolate the effect of feature X_j on predictions

Suppose we have 3 observations and want PD for DTI ratio:

Obs	DTI	Income	Rate	...	Original Prediction
1	20%	\$50k	12%	...	$P(\text{def}) = 0.15$
2	35%	\$80k	15%	...	$P(\text{def}) = 0.25$
3	25%	\$45k	10%	...	$P(\text{def}) = 0.18$

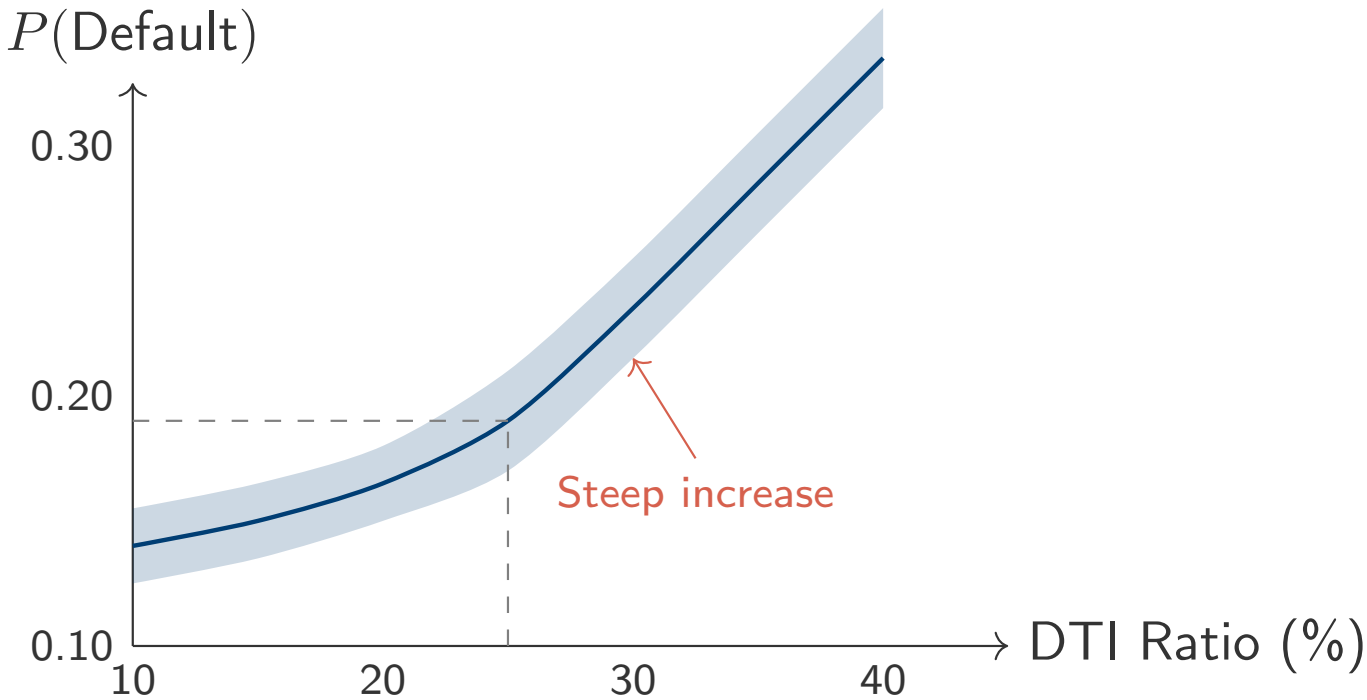
To compute PD at DTI = 30%: Set DTI to 30% for *all* observations

Obs	DTI	Income	Rate	...	New Prediction
1	**30%**	\$50k	12%	...	$P(\text{def}) = 0.22$
2	**30%**	\$80k	15%	...	$P(\text{def}) = 0.20$
3	**30%**	\$45k	10%	...	$P(\text{def}) = 0.24$

$$\text{PD}(\text{DTI} = 30\%) = \frac{0.22 + 0.20 + 0.24}{3} = 0.22$$

Repeat for DTI = 10%, 15%, 20%, ..., 50% to build the full curve.

Partial Dependence: Visualization

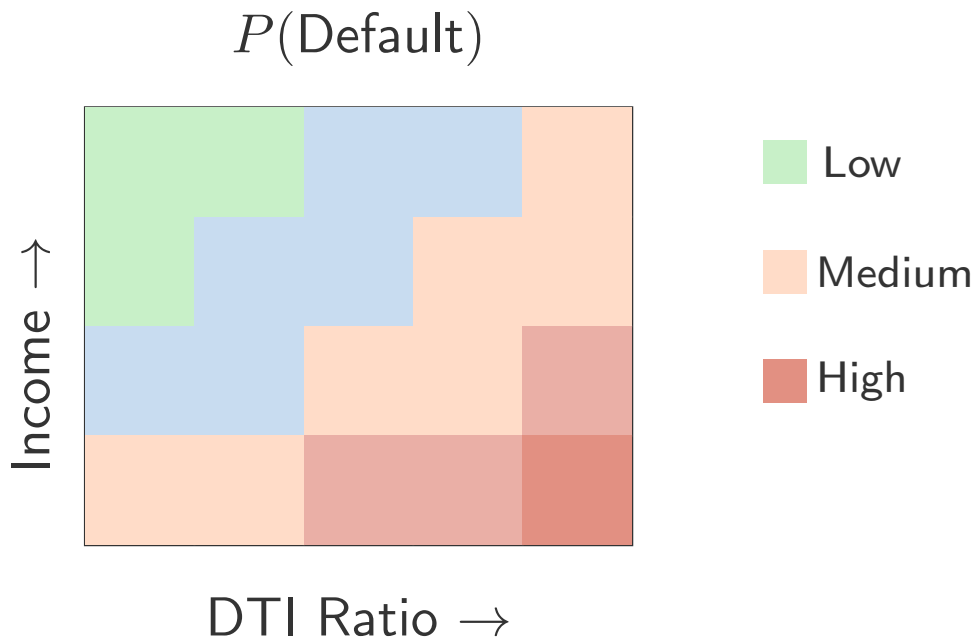


Interpretation:

- Default probability increases with DTI ratio (as expected)
- Relationship is **non-linear**: steep increase after DTI $> 25\%$
- Shaded area shows variability across the data

Two-Way Partial Dependence

Can also show interactions between two features:



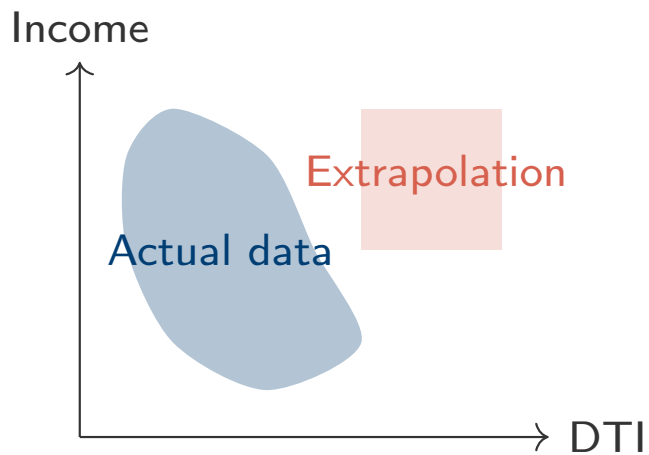
Shows interaction: High DTI is less problematic for high-income borrowers (can afford the debt service).

PDP Limitations: The Independence Assumption

PDP assumes features are independent.

Problem: When computing PD for DTI $= 50\%$, we combine it with *all* observed income levels, including very high incomes.

But in reality, high DTI and high income rarely occur together!



Result: PDP may show predictions for **unrealistic** feature combinations.

SHAP Values

The Gold Standard: SHAP

SHAP = SHapley Additive exPlanations

What SHAP provides:

- **Local** explanation for each prediction
- Additive: contributions sum to prediction
- Based on game theory (Shapley values)
- Theoretically grounded with desirable properties

The key question SHAP answers:

> *"For this specific prediction, how much did each feature contribute to pushing the prediction above or below the average?"*

Shapley Values: Game Theory Foundation

Origin: Nobel Prize-winning concept from cooperative game theory (Lloyd Shapley, 1953).

The game theory analogy:

- **Players** = Features
- **Game** = Prediction task
- **Payout** = Prediction minus average prediction
- **Question:** How to fairly divide the "payout" among players?

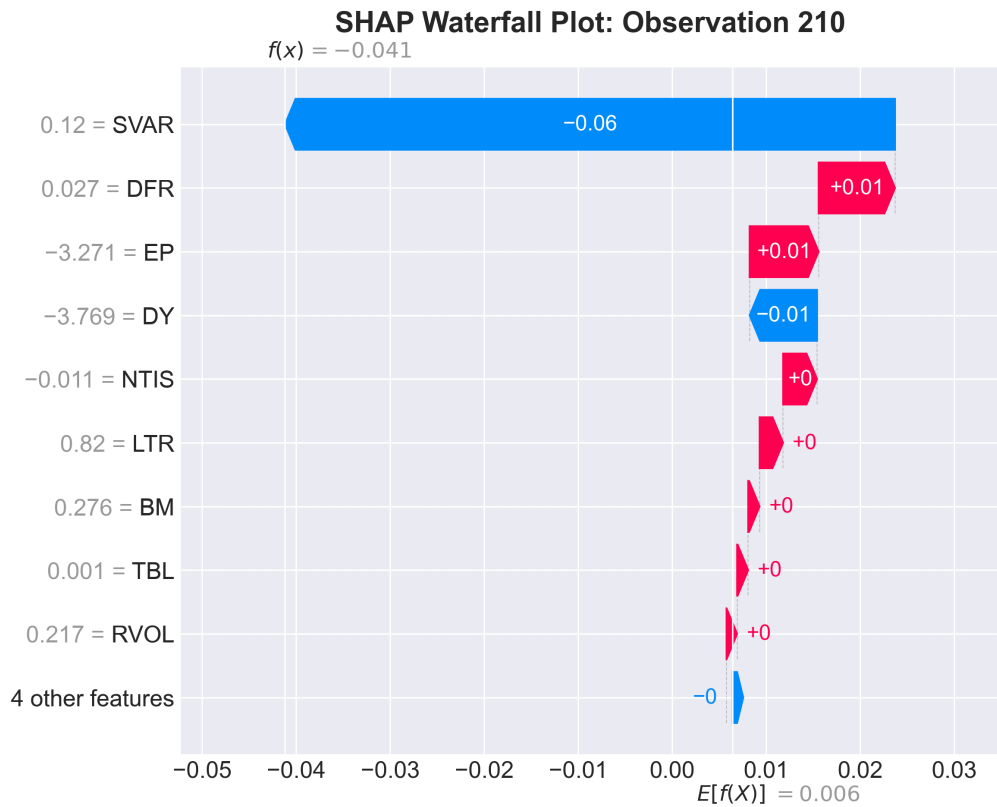
Shapley's answer: Give each player their **average marginal contribution** across all possible coalitions (orderings).

Shapley Value: Intuition

Example: ERP forecast

- Average prediction: 0.006
- January 2020 prediction: -0.041
- Difference to explain: $0.006 - (-0.041) = 0.047$

SHAP decomposes this as:



High SVAR (-0.06) push prediction down. High default risk $(+0.01)$ and earnings-price ratio $(+0.01)$ provide minimal offset.

SHAP: Formal Definition

For feature j and observation x:

$$\phi_j(x) = \sum_{S \subseteq \{1,\ldots,p\} \setminus \{j\}} \frac{|S|!(p - |S| - 1)!}{p!} \left[f(S \cup \{j\}) - f(S) \right]$$

In words:

- Consider all possible subsets S of features (excluding j)
- For each subset, compute the marginal contribution of adding j
- Weight by the number of orderings that produce this subset
- Average across all subsets

Key Property

SHAP values are **additive**: $f(x) = \phi_0 + \sum_{j=1}^{p} \phi_j(x)$
where ϕ_0 is the average prediction.

Why SHAP? Desirable Properties

SHAP is the *only* method satisfying all of:

1. **Local accuracy:** Contributions sum to prediction

$$f(x) = \phi_0 + \sum_{j=1}^{p} \phi_j(x)$$

2. **Missingness:** Features not in the model get $\phi_j = 0$
3. **Consistency:** If a feature's contribution increases in the model, its SHAP value doesn't decrease
4. **Symmetry:** Features that contribute equally get equal SHAP values

Theoretical Foundation

These properties uniquely determine the Shapley value formula.

Computing SHAP: The Computational Challenge

Problem: Exact Shapley values require summing over 2^p subsets!

With 20 features: $2^{20} \approx 1$ million subsets per prediction.

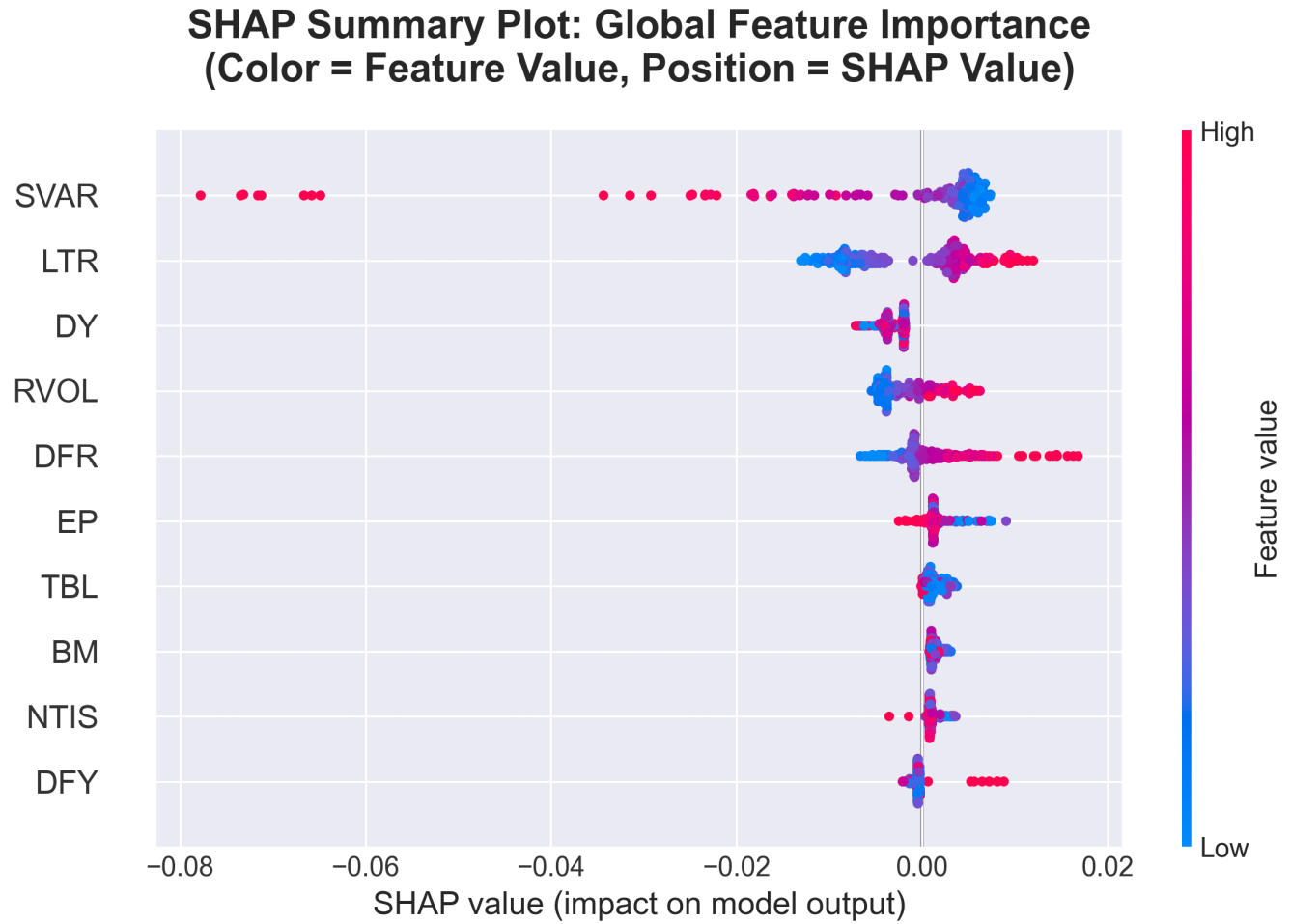
Solutions:

Method	Description
TreeSHAP	Exact, fast algorithm for tree-based models. $O(TLD^2)$ complexity.
KernelSHAP	Model-agnostic approximation using weighted regression. Slower but works for any model.
DeepSHAP	Approximation for neural networks using DeepLIFT.

Practical Advice

Use `TreeSHAP` for tree-based models. Use `KernelSHAP` for other models.

SHAP Summary Plot: Global View



Reading: High volatility (red dots on left) decreases expected returns. Other predictors have mixed effects.

SHAP: Advantages and Limitations

Advantages:

- Solid theoretical foundation (game theory)
- Local *and* global interpretation
- Additive: contributions sum to prediction
- Rich visualization options

Limitations:

- It can be slow for large datasets
- Assumes feature independence (like PDP)
- Can be sensitive to background data choice
- May be complex to explain to non-technical stakeholders

Best Practice

SHAP is the go-to method for model interpretation in practice. Use TreeSHAP for tree models, supplement with simpler explanations for stakeholders.

Motivation: Investment Models

Why Interpretability Matters for Investment Models

Interpretability matters above and beyond regulatory requirements

1. **Model validation**
   - Do feature effects match economic intuition?
   - Are we capturing signal or noise?
2. **Regime detection**
   - What's driving the current forecast?
   - Has the market regime changed?
3. **Risk management**
   - Why is the model bullish/bearish?
   - What could make the forecast wrong?
4. **Client communication**
   - Explaining investment decisions to stakeholders
   - Building trust in quantitative strategies

The Equity Premium Prediction Problem

Goal: Predict next month's excess return on the stock market

$$r^e_{t+1} = r^{mkt}_{t+1} - r_f$$

Why might returns be predictable?

- **Time-varying risk premia:** Expected returns compensate for risk, and risk varies over the business cycle
- **Behavioral biases:** Investor overreaction, sentiment, slow information diffusion
- **Limits to arbitrage:** Mispricing persists because arbitrage is costly and risky

The academic debate:

- Goyal & Welch (2008): Most predictors fail out-of-sample
- Campbell & Thompson (2008): Economically meaningful even with low R^2
- Gu, Kelly & Xiu (2020): ML methods improve upon linear models

Key Predictor Variables: Details

Variable	Name	Economic Rationale
`d_p`	Dividend-price ratio	High $\Rightarrow$ cheap market
`d_y`	Dividend yield	Income return component
`e_p`	Earnings-price ratio	Valuation (inverse P/E)
`b_m`	Book-to-market	Value indicator
`tbl`	T-bill rate	Monetary policy
`tms`	Term spread	Recession probability
`dfy`	Default spread	Credit risk premium
`svar`	Stock variance	Market uncertainty
`ntis`	Net equity issuance	Corporate timing signal
`infl`	Inflation	Real return erosion

Plus: `skvw` (skewness), `avgcor` (correlation), `dtoy` (turnover) — behavioral/technical indicators

Target Variable: Market Excess Return

Statistic	Value
Mean	0.64%
Std Dev	4.22%
Min	−22.09%
Max	16.19%
Median	0.93%

Key observations:

- Average monthly excess return: 0.64% ($\approx$ 7.7% annualized)
- High volatility: 4.22% monthly ($\approx$ 14.6% annualized)
- Left tail: worst month was −22% (October 1987)
- **Signal-to-noise ratio is low!**

Time-Series Train/Test Split

Critical: Use chronological split, not random!

```
# Define features and target
features = [c for c in df.columns if c not in ['DATE', 'MktRf']]
X = df[features].values
y = df['MktRf'].values

# Time-series split:  first 70% for training
split_idx = int(len(df) * 0.7) # = 577
X_train, X_test = X[:split_idx], X[split_idx:]
y_train, y_test = y[:split_idx], y[split_idx:]

print(f"Training:  1953-04 to 2001-04 (n=577)")
print(f"Testing:  2001-05 to 2021-12 (n=248)")
```

Why Chronological Split?

Random splits cause **look-ahead bias** — the model sees "future" data during training. This inflates apparent performance.

Training the Random Forest

```
# Train Random Forest
rf = RandomForestRegressor(
n_estimators=500,
max_depth=5, # Shallow trees to avoid overfitting
random_state=42,
n_jobs=-1
)
rf.fit(X_train, y_train)

# Evaluate
from sklearn.metrics import r2_score
y_pred_train = rf.predict(X_train)
y_pred_test = rf.predict(X_test)

print(f"In-sample R2:  {r2_score(y_train, y_pred_train):.4f}")
print(f"Out-of-sample R2:  {r2_score(y_test, y_pred_test):.4f}")
```

Results:

- In-sample R^2: **0.5811** (58% variance explained)
- Out-of-sample R^2: **0.1365** (14% variance explained)

Model Performance Discussion

In-sample $R^2 = 0.58$ vs. Out-of-sample $R^2 = 0.14$

Is this good?

- For equity premium prediction: **Yes!**
- Most linear models achieve OOS $R^2 < 0$ (worse than mean)
- Campbell & Thompson (2008): even $R^2 = 0.5\%$ monthly is economically significant

The gap between in-sample and out-of-sample:

- Some overfitting, but not catastrophic
- Shallow trees (max_depth=5) help regularize
- Test period (2001–2021) includes 2008 crisis, COVID — challenging!

Key Question

What's driving the predictions? Let's use interpretability methods to find out.

Economic Significance vs. Statistical Significance

Why does $R^2 = 14\%$ matter economically?

Consider a mean-variance investor allocating between stocks and bonds:

$$\omega_t = \frac{1}{\gamma} \cdot \frac{\mathbb{E}_t[r^e_{t+1}]}{\sigma^2}$$

With predictability:

- Can time the market: increase weight when expected returns are high
- Utility gain from timing $\propto R^2 \times$ Sharpe ratio^2

Back-of-envelope calculation:

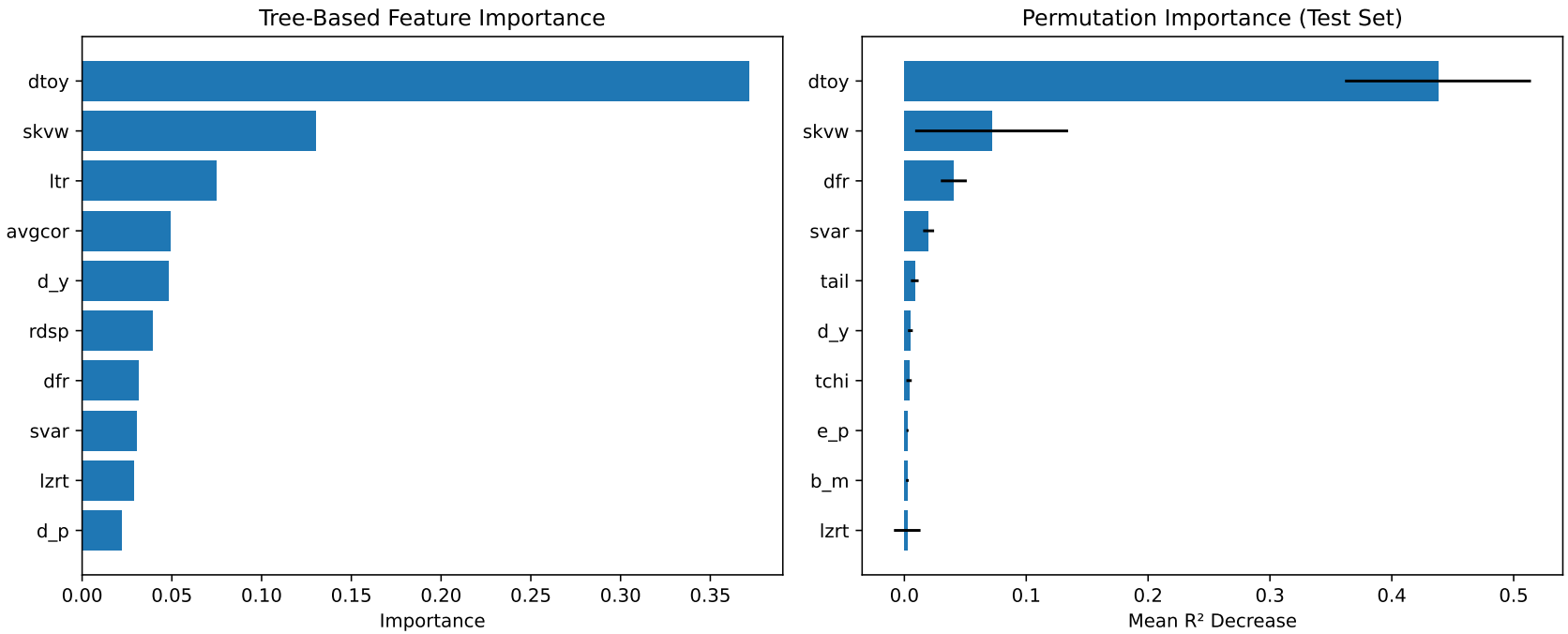
- $R^2 = 14\% \Rightarrow$ correlation ≈ 0.37 between forecast and realized
- Annualized Sharpe improvement: ≈ 0.15–0.25
- For a \$1B portfolio: potentially \$15–25M in alpha per year

Caveat

These gains assume no transaction costs, perfect execution, and stable relationships!

Opening the Black-Box

Feature Importance



Left: Tree-based (in-sample). **Right:** Permutation (out-of-sample). `dtoy` dominates in both!

Key Observations from Feature Importance

Feature	Tree-Based Rank	Permutation Rank
dtoy	1	1
skvw	2	2
dfr	7	3
svar	8	4
ltr	3	10
avgcor	4	11

Key observations:

- `dtoy` and `skvw` are top 2 in both methods — **robust finding**
- `dfr` (default spread) and `svar` (volatility) rank higher in permutation
  - These capture crisis/stress periods (important OOS)
- `ltr` and `avgcor` may be overfit in-sample

Recommendation: Trust permutation importance for model validation.

Economic Interpretation of Feature Importance

Surprising result: Traditional valuation ratios (d_p, e_p, b_m) rank low!

What dominates instead?

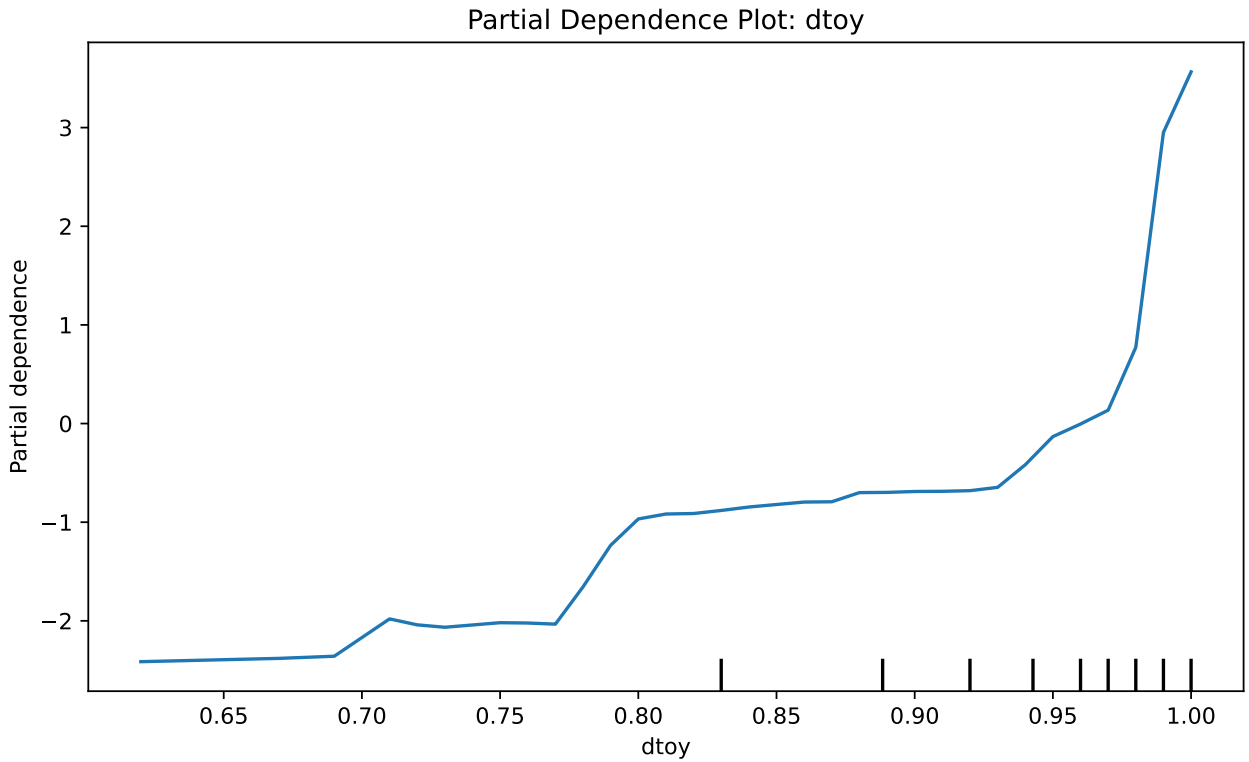
1. **dtoy (Detrended Turnover):** Behavioral/sentiment indicator
   - High turnover = high trading activity = investor enthusiasm
   - Related to Baker & Wurgler's sentiment measures
2. **skvw (Skewness):** Crash risk / tail risk
   - Investors dislike negative skewness (lottery preferences)
   - High crash risk $\Rightarrow$ investors demand higher returns
3. **dfr, svar:** Stress/volatility indicators
   - Capture time-varying risk aversion
   - Counter-cyclical: high in crises, low in booms

Implication

The model relies more on **behavioral/risk** variables than on **valuation** ratios. This suggests returns are driven by sentiment and risk premia, not just mean reversion.

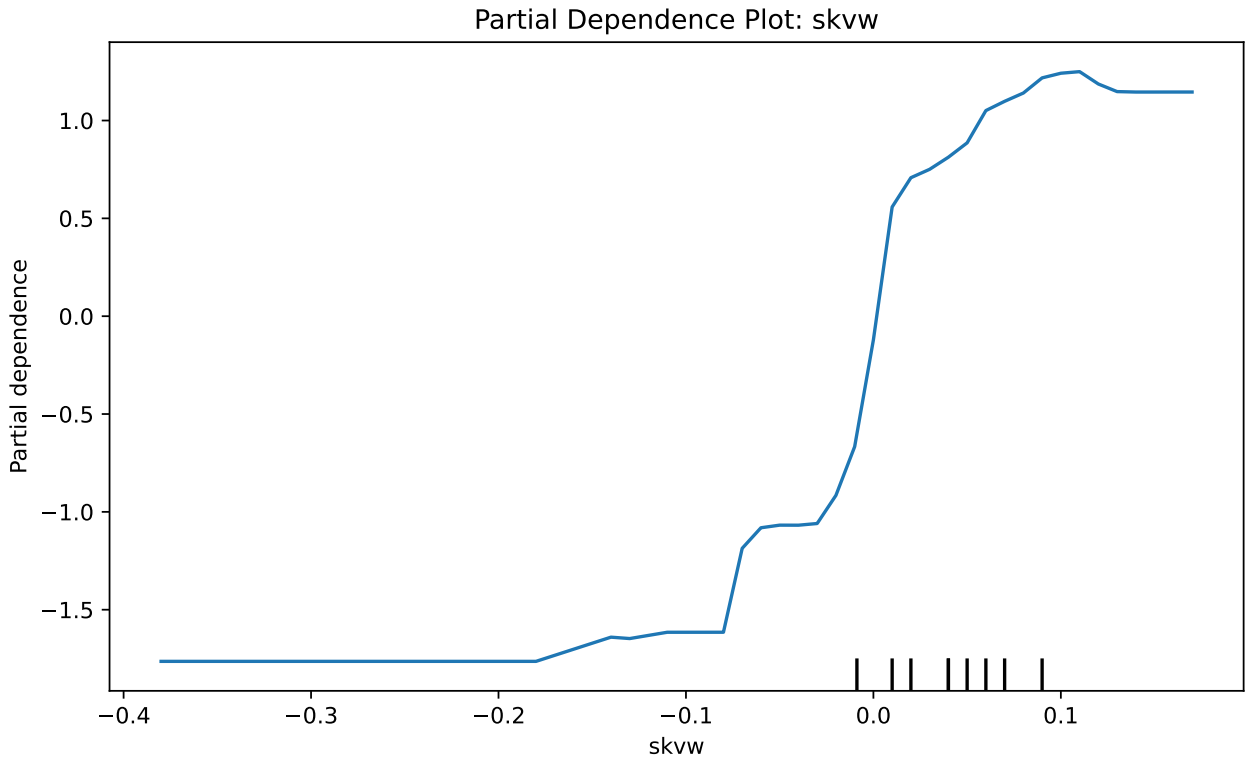
PDP: dtoy (Detrended Turnover)

`dtoy`: Detrended turnover — measures trading activity relative to trend. High values indicate unusual trading volume.



Interpretation: Low turnover → bearish. High turnover → very bullish. **Strongly nonlinear** effect above dtoy = 0.97.

PDP: skvw (Market Skewness)



Interpretation: Negative skewness (crash risk) $\rightarrow$ bearish. Positive skewness $\rightarrow$ bullish. Effect is monotonic but saturates.

Economic Interpretation of PDPs

dtoy (Detrended Turnover) — The Sentiment Channel:

- **High turnover:** Investors actively trading, optimism, momentum
- **Low turnover:** Apathy, fear, or lack of conviction
- Academic link: Baker & Stein (2004) — liquidity as sentiment proxy

skvw (Market Skewness) — The Tail Risk Channel:

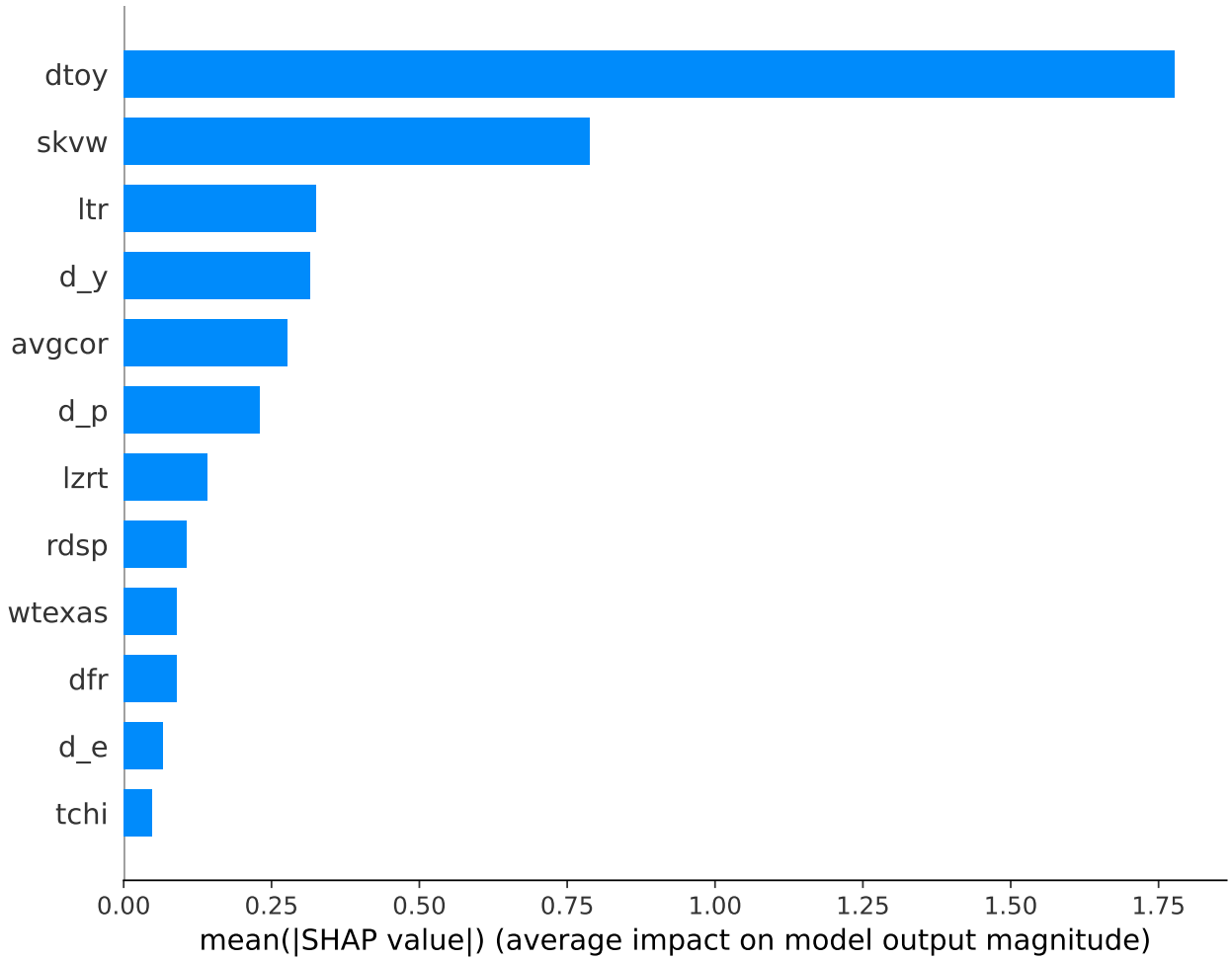
- **Negative skewness:** Left tail risk (crash potential)
- Investors are averse to negative skewness (Barberis & Huang, 2008)
- When crash risk is high, investors demand higher expected returns

Key Insight

The model captures **time-varying risk premia**: when investors are scared (low turnover, high crash risk), they require higher compensation $\Rightarrow$ expected returns rise.

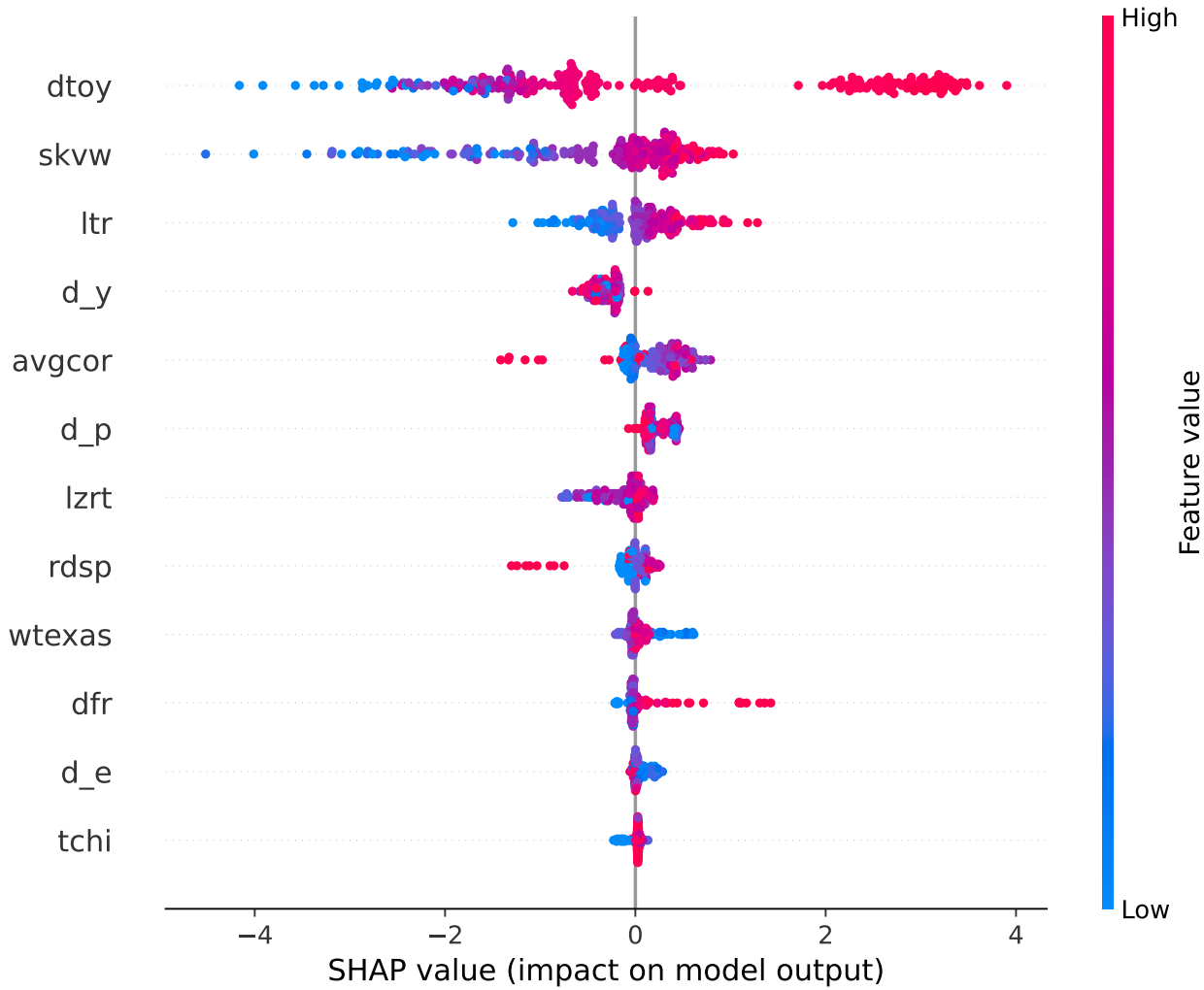
SHAP Analysis

SHAP Summary Plot: Bar Chart



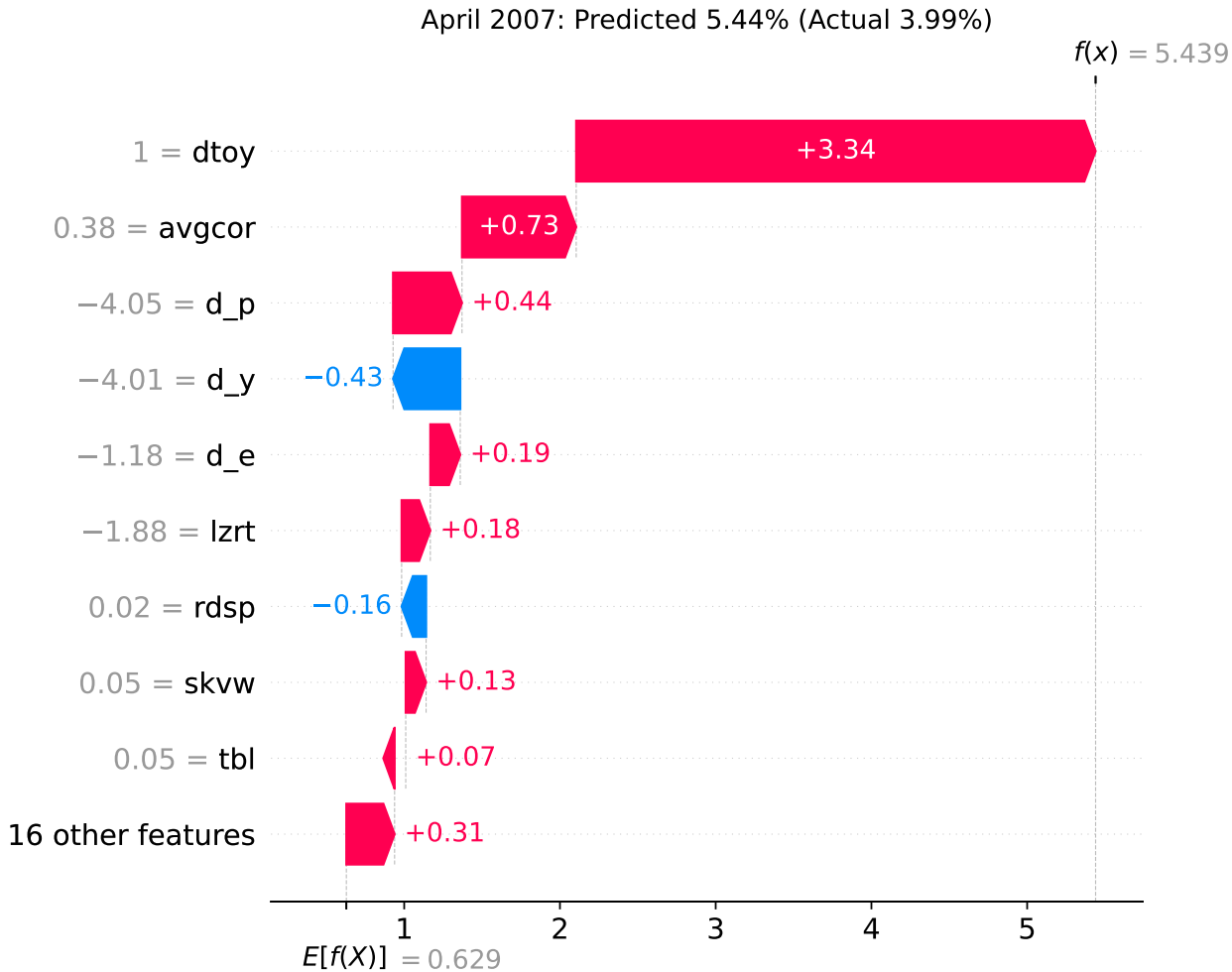
Interpretation: Mean |SHAP| shows average impact. `dtoy` contributes ± 1.78 percentage points on average — huge given baseline is only 0.63%!

SHAP Summary Plot: Beeswarm



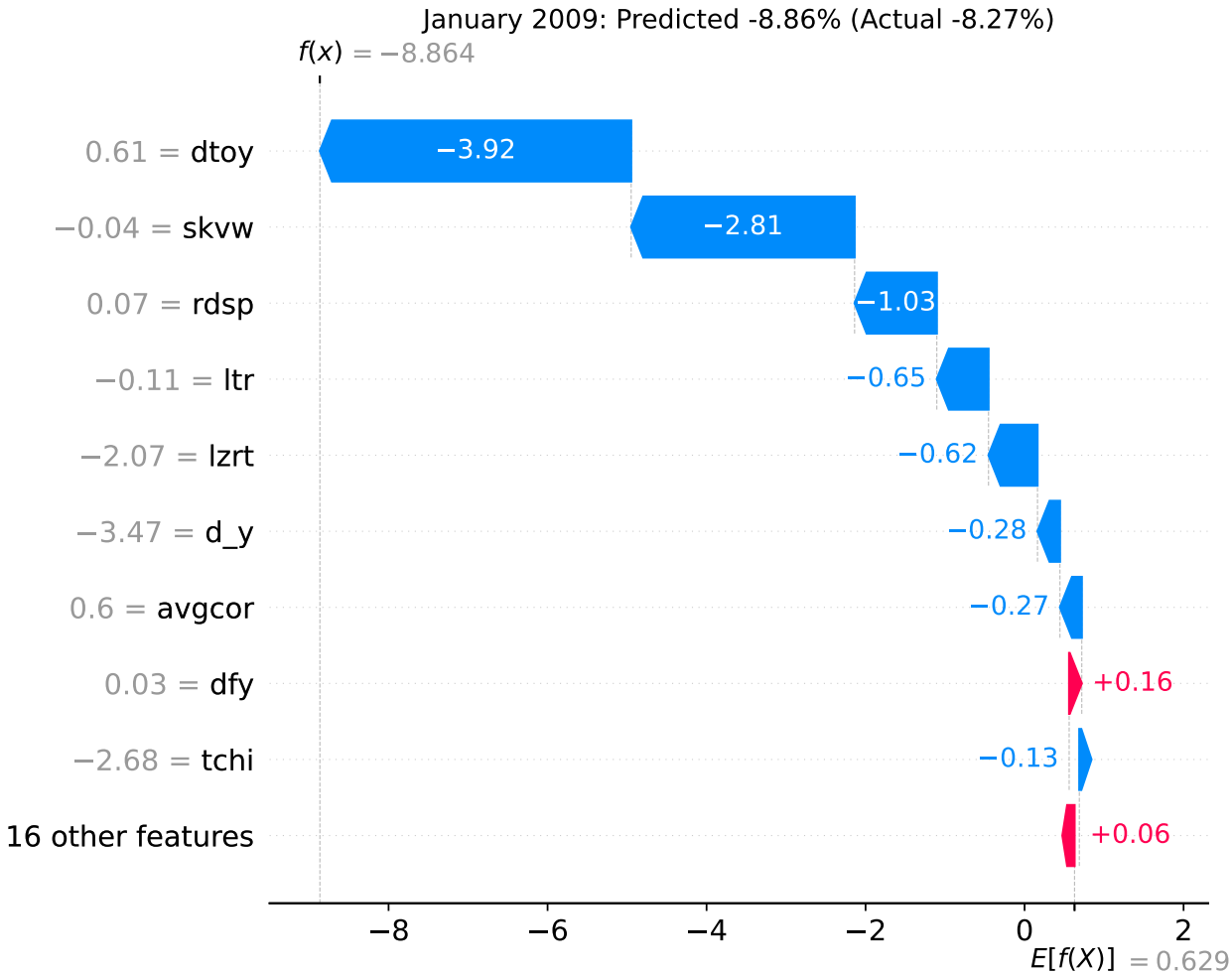
Each dot = one observation. Color = feature value (red=high, blue=low). High `dtoy` $\rightarrow$ positive SHAP (bullish).

SHAP Waterfall: April 2007 (Bullish Signal)



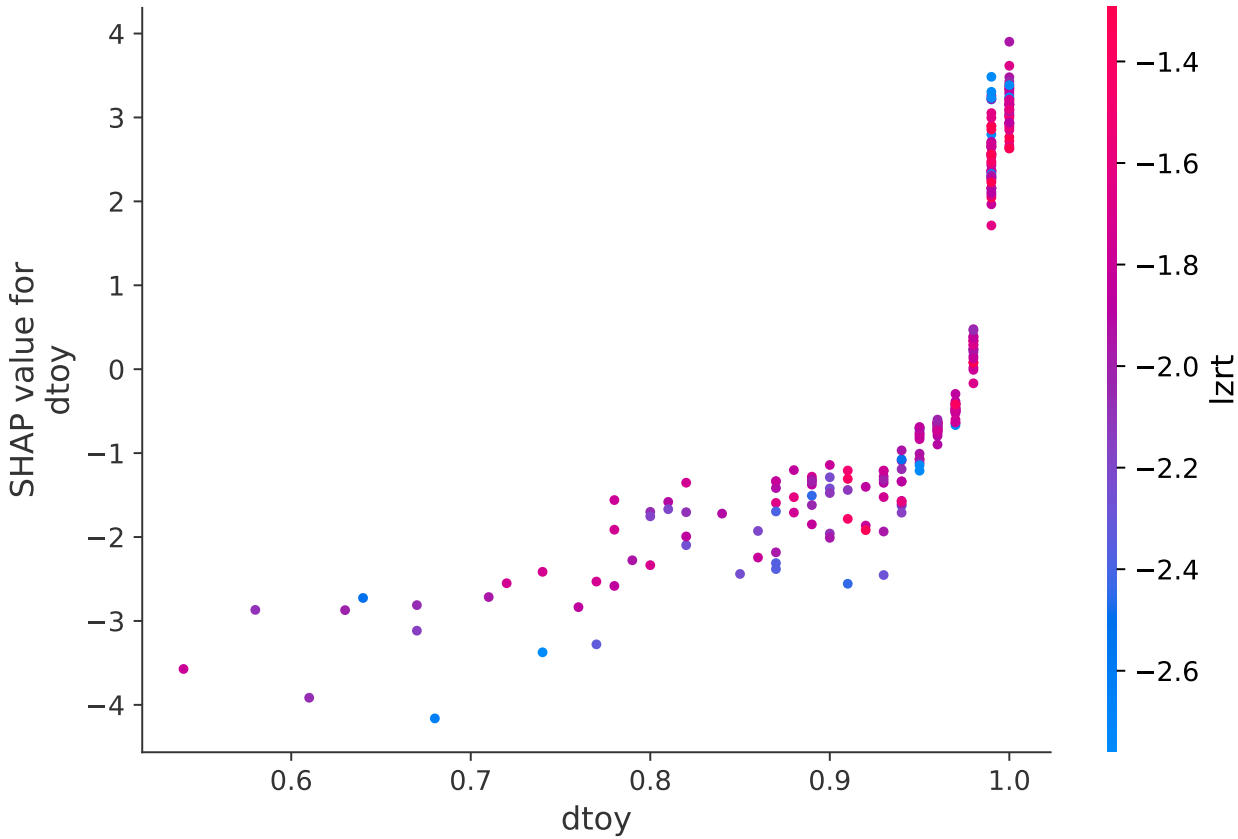
Why bullish? Turnover at maximum (dtoy=1.00) pushed prediction up by +3.34 percentage points!

SHAP Waterfall: January 2009 (Bearish Signal)



Why bearish? Low turnover, negative skewness, credit stress. Model correctly identified the financial crisis!

SHAP Dependence Plot: dtoy



SHAP value vs. feature value reveals the nonlinear relationship. Color shows interaction with another feature.

Validating the Model: Do Effects Match Theory?

SHAP reveals the effects direction: do they make economic sense?

Feature	High $\rightarrow$	Theory	Mechanism
dtoy	Bullish	Behavioral	Sentiment/momentum
skvw	Bullish	Risk-based	Low crash risk
avgcor	Bullish	Behavioral	Herding/optimism
dfr	Bearish	Risk-based	Credit stress
svar	Bearish	Risk-based	Uncertainty
d_y	Bullish	Valuation	Mean reversion

All effects align with theory! This increases confidence that:

- The model captures real economic relationships, not spurious patterns
- Predictions can be trusted when economic conditions match training data
- We can anticipate when the model might fail (regime changes)

Case Study: The 2007–2009 Financial Crisis

Does the model capture known economic events?

April 2007 (Pre-crisis peak):

- dtoy = 1.00 (maximum turnover) — "irrational exuberance"
- avgcor = 0.38 (high) — herding behavior
- Model prediction: +5.44% (bullish)
- **Economic interpretation:** Sentiment-driven rally, classic late-cycle behavior

January 2009 (Crisis trough):

- dtoy = 0.61 (low turnover) — fear, paralysis
- skvw = −0.04 (negative) — crash risk elevated
- dfr elevated — credit markets frozen
- Model prediction: −8.86% (bearish)
- **Economic interpretation:** Flight to quality, deleveraging, fire sales

Insight

Interpretability lets us **tell an economic story** about each prediction!

Portfolio Implications

From Prediction to Portfolio

How would an investor use this model?

Simple market timing strategy:

- If $\hat{r}^e_{t+1} > \bar{r}$: Overweight equities
- If $\hat{r}^e_{t+1} < \bar{r}$: Underweight equities (or go to cash)

Interpretability helps with:

1. **Position sizing:** Large positions only when drivers are clear
2. **Risk management:** Know which factors could reverse the signal
3. **Regime awareness:** Recognize when the model is in/out of regime
4. **Client communication:** Explain why you're bullish/bearish

Example

"We're overweight equities because market turnover is elevated and crash risk is low. Key risk: if credit spreads widen, our signal would flip bearish."

When to Trust (and Distrust) the Model

Trust the model when:

- SHAP decomposition tells a coherent economic story
- Dominant features (dtoy, skvw) are the main drivers
- Current regime resembles training data

Be cautious when:

- Prediction is driven by minor/unstable features
- Features are at extreme values (extrapolation)
- Market structure has changed (e.g., rise of passive investing)
- SHAP contributions nearly cancel out (low conviction)

Risk Management Principle

Use SHAP to compute "conviction scores" — size positions based on how cleanly the prediction is explained by theoretically-grounded factors.

Limitations and Model Risk

Every model has limitations — interpretability helps identify them:

1. **Structural breaks:**
   - Model trained 1953–2001 may not capture post-GFC dynamics
   - Rise of algorithmic trading, passive investing, QE
2. **Crowding:**
   - If everyone uses the same predictors, alpha disappears
   - GWZ predictors are well-known in academia
3. **Nonlinear extrapolation:**
   - PDP shows extreme nonlinearity for dtoy > 0.97
   - What happens at dtoy $= 1.05$? (Never seen in training)
4. **Omitted variables:**
   - No Fed policy, no geopolitical risk, no ESG factors

Practical Workflow

Interpretability Workflow for Investment Models

Step 1: Train model with proper validation

- Time-series split (no look-ahead bias)
- Report in-sample AND out-of-sample metrics

Step 2: Feature importance analysis

- Compare tree-based and permutation importance
- Flag features that rank differently (potential overfit)

Step 3: Understand feature effects (PDP)

- Are effects monotonic? Nonlinear? Threshold effects?
- Do they match economic intuition?

Step 4: Explain individual predictions (SHAP)

- What's driving today's forecast?
- Examine extreme predictions carefully

Model Documentation Checklist

For any deployed investment model, document:

Item	Completed
In-sample and OOS performance metrics	☐
Time-series validation methodology	☐
Feature importance (both methods)	☐
PDP for top 5 features	☐
SHAP summary plot	☐
Example SHAP force plots (bull/bear)	☐
Economic interpretation of effects	☐
Known limitations and failure modes	☐
Monitoring plan	☐

This documentation supports model governance, risk management, and client communication.

Summary

Key Takeaways

1. **Interpretability validates economic content**
   - Feature effects should match finance theory
   - Our model: sentiment (dtoy) and risk (skvw, dfr) dominate
2. **Risk vs. Behavioral channels**
   - Traditional valuation ratios (d_p, e_p) are NOT the top predictors
   - Behavioral/sentiment variables dominate — supports limits to arbitrage
3. **SHAP enables economic storytelling**
   - April 2007: Sentiment-driven late-cycle optimism
   - January 2009: Fear, crash risk, credit stress
4. **Practical value for portfolio management**
   - Position sizing based on explanation quality
   - Risk management: know what could flip the signal
   - Client communication: tell coherent economic stories

Readings and Next Steps

Readings for Today's Lecture:

- Goyal & Welch (2008), "Equity Premium Prediction," *RFS* *[Supp.]*
- Gu, Kelly & Xiu (2020), "Empirical Asset Pricing via ML," *RFS* *[Supp.]*
- Baker & Wurgler (2006), "Investor Sentiment," *JF* *[Supp.]*

Coming Up Next Week:

- Introduction to neural networks
  - Feedforward neural networks: architecture, activation
  - Backpropagation and gradient descent
  - Regularization: dropout, early stopping, batch normalization
- Financial applications: return prediction, default risk

Questions?