Supervised Learning: Regressions

In **supervised learning**, an algorithm is provided historical data (both input and output variables), and is trying to find the relationship that has the best predictive power on out-of-sample data. Methods of supervised learning are further classified in methods of regression and methods of classification. **Regressions** try to predict output variables based on a number of input variables. **Classification** methods attempt to group or classify output into categories. For instance we may want the output of a model to be a binary action as 'buy' or 'sell' based on a number of variables. One can think of regression and classification as the same methods, with the forecast of a regression being a continuous number (e.g. market will go up 1%, 1.15%, 2%, etc.) and forecast of classification being a discrete number (e.g. buy=1, sell=0, or volatility regime will be: high=+1, medium=0, low=-1).

Even a simple linear regression can be thought of as a supervised Machine Learning method. However, linear regressions may not be suitable to deal with outliers, a large number of variables, variables that are correlated, or variables that exhibit non-linear behavior. To illustrate one example of the inadequacy of simple linear regression, consider a hypothetical regression forecast of a risky asset price:

Asset Price = 0.2*US Growth - 0.05*EM Growth + 0.1* US HY Bonds + 22* US Equities – 21* Global Equities

Ordinary Linear regression[29] typically produces these types of results (i.e. large opposite sign coefficients) when one includes variables that are highly correlated (such as US and Global Equities). In Big Data strategies, one is expected to include a large number of variables, without necessarily knowing which ones are correlated and which ones are not. The simple regression above would suggest an irrational trading strategy with unnecessarily high leverage and idiosyncratic risk. One simple extension of linear regression is called a **Lasso regression**. Lasso regression tries to establish the relationship (a forecast) by choosing the smallest and most relevant set of input variables. In the example above, Lasso regression would 'realize' that US and Global Equities are highly correlated, and would penalize and eliminate the large long-short position.

K-nearest neighbors method forecasts data by simply looking at historical samples and establishing what has happened in similar situations as a best forecast of the future. While K-nearest neighbors is a simplistic method that overly relies on historical patterns, it has the advantage of including non-linear effects. In this section, we analyze the following regression methods: **Lasso, Ridge, Elastic Net, and K-nearest neighbors**.

In the Mathematical box below we provide some additional theory behind regressions in the context of supervised Machine Learning. This can be skipped by all but the most quantitatively inclined readers.

In supervised learning, we seek to determine an appropriate functional relationship between an output or target variable and a set of input or predictor variables. Given m training examples, denoted by $(\underline{x}^{(i)}, y^{(i)})$, we seek the function h such that $y = h(\underline{x})$. Defining $\mathcal{X}$ as the input space for input variables (say, $\mathbb{R}^n$) and $\mathcal{Y}$ as the space of output values (say, $\mathbb{R}$); it is conventional to refer to the function $h: \mathcal{X} \rightarrow \mathcal{Y}$ as the hypothesis function. The learning task is called either a regression or a classification, depending on whether the output space $\mathcal{Y}$ is continuous (as in $\mathbb{R}$) or discrete (as in $\{0, 1\}$).

In the classical Ordinary Least Squares model for linear regression, one defines $h_{\underline{\theta}}(\underline{x}) = \underline{\theta}^T \underline{x}$, where $\underline{\theta}$ is the parameter or weight vector. Errors during model fitting are penalized using the cost function $J(\underline{\theta}) = \frac{1}{2} \sum_{i=1}^{m} \left(h_{\underline{\theta}}(\underline{x}^{(i)}) - y^{(i)}\right)^2$. The

[29] For classical extensions and analysis of ordinary linear regression, see Seber (1984), Weisberg (1980) and Kennard (1970); for ridge, see Hoerl and Kennard (1970); for lasso, see Tibshirani (1996). Shrinkage methods are compared in Frank and Friedman (1993). Bayesian methods for estimating regression parameters are covered in George and McCulloch (1993), Madigan and Raftery (1994), Clyde, DeSimone and Parmigiani (1996), West (2003).Basis pursuit in signal processing was proposed in Chen et al (1998). For least angle regression and related homotopy methods, see Efron et al (2004), Osborne et al (2000a) and Osborne et al (2000b). Forward stage-wise criterion and a path algorithm for generalized regression models are covered in Hastie et al (2007) and Park and Hastie (2007). PLS (Partial least squares) was introduced in Wold (2007).

traditional approach was to stack the training examples to form $X = \begin{bmatrix} \underline{x}^{(1)T} \\ \vdots \\ \underline{x}^{(m)T} \end{bmatrix}$ and $\underline{y} = [y^{(1)} \quad \cdots \quad y^{(m)}]^T$. Matrix differentiation of the cost function $J(\underline{\theta}) = || X\underline{\theta} - \underline{y} ||^2$ yields the normal equations $X^T X\underline{\theta} = X^T \underline{y}$ and the estimator as $\underline{\theta} = (X^T X)^{-1} X^T \underline{y}$. Unfortunately, this traditional approach of theoretical derivation and practical use is not extensible to modern Machine Learning models. So we review this question using two other approaches, namely

a. A purely numerical approach, which we shall use to derive perceptron models; and
b. A purely probabilistic approach, which we shall use to derive ridge and lasso regressors.

The numerical approach is to note that $J(\underline{\theta})$ can be minimized using the gradient descent algorithm, where the j[th] element of the vector $\underline{\theta}$ is updated as $\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\underline{\theta})$. Here α is the learning rate. Repeated decrease in the direction of steepest descent of $J(\underline{\theta})$ leads to convergence to a local (global in the case of convex functions) minima. Applied to our definition of $J(\underline{\theta})$ and denoting the j[th] element of the vector $\underline{x}^{(i)}$ as $x_j^{(i)}$, the gradient descent rule yields $\theta_j \leftarrow \theta_j + \alpha \left(y^{(i)} - h_{\underline{\theta}}(\underline{x}^{(i)})\right) x_j^{(i)}$. This rule is called the Widrow-Hoff or Least Mean Squares (LMS) rule in Machine Learning literature.

When the LMS rule is applied – as shown below - in one shot to all training examples, it is called Batch Gradient Descent.

Repeat until convergence
{ $\forall j \in \{1, \ldots, n\}, \ \theta_j \leftarrow \theta_j + \alpha \sum_{i=1}^{m} \left(y^{(i)} - h_{\underline{\theta}}(\underline{x}^{(i)})\right) x_j^{(i)}$ }

If the training set is very large, we can update all the weights using information from each individual training example, instead of iterating over the entire training set to update a single weight parameter. This idea leads to Incremental or Stochastic Gradient Descent (SGD).

Repeat until convergence
{ $\forall i \in \{1, \ldots, m\}$ { $\forall j \in \{1, \ldots, n\}, \ \theta_j \leftarrow \theta_j + \alpha \left(y^{(i)} - h_{\underline{\theta}}(\underline{x}^{(i)})\right) x_j^{(i)}$ } }

While stochastic gradient descent lacks theoretical guarantees on convergence of $\underline{\theta}$ to the local minima, one finds its performance to be superior to batch gradient descent on large data sets. We shall use stochastic gradient descent to train many of the models employed in this primer. The above LMS rule shall also recur in identical form when we study logistic classifiers.

To motivate and understand many of the Machine Learning algorithms, researchers rely frequently on probabilistic interpretations. In the context of Ordinary Least Squares, one can model the output as $y^{(i)} = \underline{\theta}^T \underline{x}^{(i)} + \varepsilon^{(i)}$. If $\varepsilon^{(i)} \sim N(0, \sigma^2)$ represents independent and identically distributed (i.i.d.) noise, then the likelihood of observing the particular training set is given by the likelihood function

$$L(\underline{\theta}) = p(y|X; \underline{\theta}) = \prod_{i=1}^{m} p(y^{(i)}|\underline{x}^{(i)}; \underline{\theta}) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left(y^{(i)} - \underline{\theta}^T \underline{x}^{(i)}\right)^2}{2\sigma^2}\right).$$

The principle of Maximum Likelihood mandates the choice of parameter $\underline{\theta}$ to maximize the above likelihood expression, i.e. render the appearance of the training set as likely as possible. Maximizing equivalently the log-likelihood $l(\underline{\theta})$, we recover

$$\hat{\theta}_{ML} = \arg\min_{\underline{\theta}} l(\underline{\theta}) = \arg\min_{\underline{\theta}} \log L(\underline{\theta}) = \arg\max_{\underline{\theta}} \frac{1}{2} || X\underline{\theta} - \underline{y} ||^2 .$$

This is same as the expression obtained in the traditional analysis for OLS. The advantage of this probabilistic modeling is that we will extend it using Bayesian priors to derive more stable linear regression techniques like ridge and lasso.

Penalized Regression Techniques: Lasso, Ridge, and Elastic Net

Penalized regression techniques like Lasso (also, spelt as LASSO) and Ridge are simple modifications of ordinary linear regression aimed at creating a more robust output model in the presence of a large number of potentially correlated variables. When the number of input features is large or when the input features are correlated, classical linear regression has a tendency to overfit and yield spurious coefficients. LASSO, Ridge, and Elastic Net regressions are also examples of *'regularization'* – a technique in Machine Learning that is expected to reduce the out-of-sample forecasting errors (but does not help with reducing in-sample backtest errors).

In ordinary linear regression, we forecast the value y to be a linear combination of the inputs $x_1,\ x_2\ , \ldots , x_n$. In short we assume that variable y has 'Betas' to a number of variables x (plus some random noise).

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \varepsilon.$$

To find the 'betas' $\beta_0, \beta_1, \ldots., \beta_n$, linear regression minimizes the historical error (square of error) between actual observations of variable 'y', and predicted (or model) values of the variable. This is the reason the method is also called least-squares (since it minimizes the square of errors):

$$\text{OLS: Minimize Historical Sum of } \left(y - \left(\beta_0 + \sum_{i=1}^{n} \beta_i x_i\right)\right)^2.$$

As we saw in the numerical example above, this minimization is not stable and can yield spurious and/or large values of betas. One way to prevent that from occurring is to change the objective function in the minimization above. Instead of minimizing the least-squares objective, we can modify it by adding a penalty term that reflects our aversion towards complex models with large 'betas'. If we change the objective to include a penalty term equal to the absolute value of the beta coefficients, i.e.

$$\text{Lasso: Minimize Historical Sum of } \left(y - \left(\beta_0 + \sum_{i=1}^{n} \beta_i x_i\right)\right)^2 + \alpha \sum_{i=1}^{n} |\beta_i|,$$

then the optimizer will set 'unnecessary' and very large betas to zero. The addition of a penalty term equal to the absolute value of the coefficients is called L_1 regularization and the modified linear regression procedure is called Lasso (or LASSO). By concentrating only on the most relevant predictors, Lasso performs an implicit feature selection for us[30].

The objective function for Lasso can be understood as follows:

- If we set $\alpha = 0$, then we recover the coefficients of ordinary linear regression.
- As α increases, we choose a smaller and smaller set of predictors, concentrating only on the most important ones

Here, α is called a parameter of the model. Similarly, if we tweak our objective to include the square of the 'betas' we arrive at Ridge regression

$$\text{Ridge: Minimize Historical Sum of } \left(y - \left(\beta_0 + \sum_{i=1}^{n} \beta_i x_i\right)\right)^2 + \alpha \sum_{i=1}^{n} \beta_i^{\,2},$$

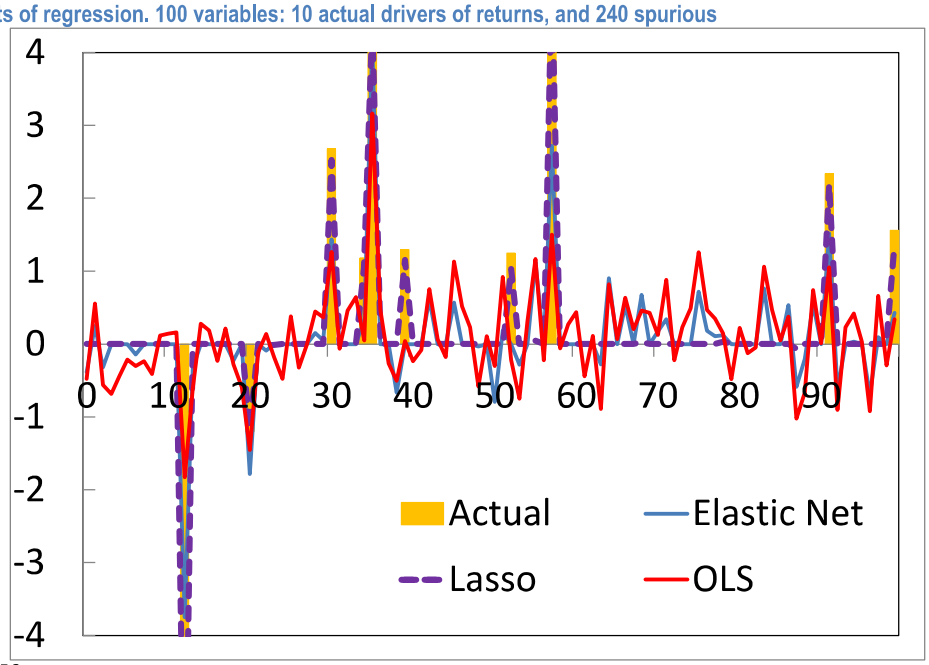
The additional penalty related to the square of the magnitude of the coefficients is called L_2 regularization and the modified linear regression procedure is called Ridge. An intermediate objective between the Ridge and Lasso objectives is used by Elastic Net regression, which finds

[30] A geometric intuition for sparsity comes from noting the level set of the L_1 norm to be a rhombus. Please see Hastie et al. (2013) for details.

$$\text{Elastic Net: Minimize Historical Sum of } \left(y - \left(\beta_0 + \sum_{i=1}^{n} \beta_i x_i\right)\right)^2 + \alpha_1 \sum_{i=1}^{n} |\beta_i| + \alpha_2 \sum_{i=1}^{n} |\beta_i|^2 .$$

We illustrate the 3 penalized regression examples with a hypothetical example (before considering an example of a realistic trading strategy). In the hypothetical example, we have chosen 10 variables that are actually driving the forecast for asset returns and added 90 spurious variables that are adding noise (throwing off the regression model). The algorithms were given all 100 variables (10 actual + 90 spurious) and asked to predict the weights for each feature. We plot the coefficients in graph below: horizontal axis ranges from 1 to 100 (reflecting the input variables) and vertical axis plots the values of beta coefficients as predicted by each algorithm. An ideal algorithm would select the 10 actual variables (light blue), and would discard the spurious ones.

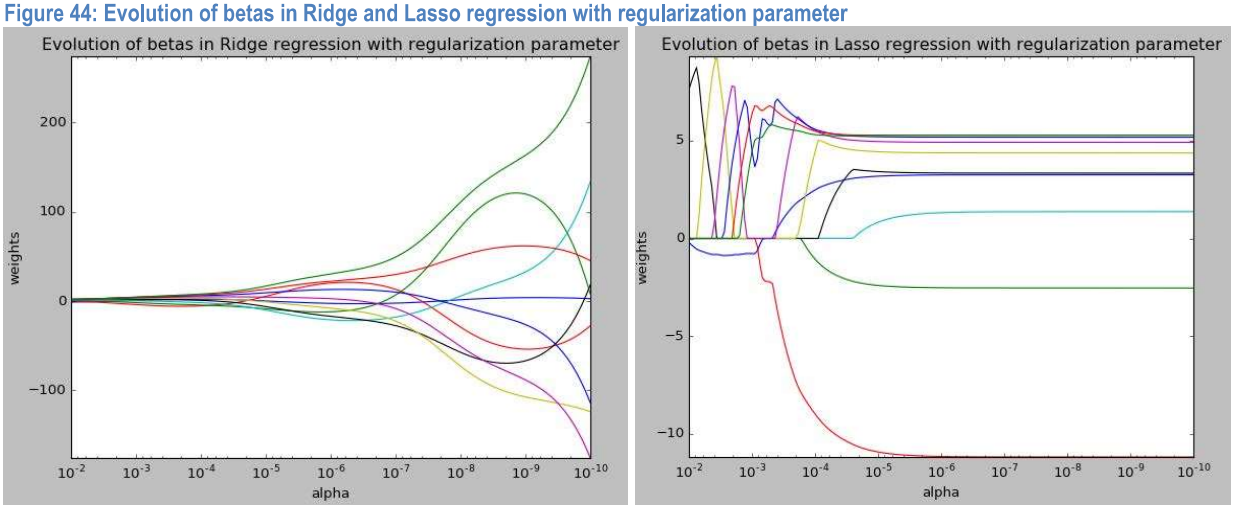
Figure 43: Coefficients of regression. 100 variables: 10 actual drivers of returns, and 240 spurious



Source: J.P.Morgan Macro QDS

Note that Lasso picked up a sub-set of actual coefficients. Ridge assigned weights to all coefficients, irrespective of whether they were spurious noise terms. Elastic net behaved in an intermediate way between Ridge and Lasso. As one increases the regularization parameter of the model (α), the models start suppressing spurious variables and focusing on actual ones. This is illustrated in the figures below showing the evolution of model 'betas' as we increase/decrease regularization parameter.

Figure 44: Evolution of betas in Ridge and Lasso regression with regularization parameter



Source: J.P.Morgan Macro QDS

Ridge, Lasso, and Elastic Net methods also have a Bayesian interpretation. While this is likely beyond the interest of practically oriented readers, we discuss it in the math box below.

Bayesian Interpretation of Penalized Regression

There is a natural Bayesian interpretation for both Ridge and Lasso regression[31]. We have shown earlier that for the linear model (assuming zero means) $\underline{y} = X\underline{\beta} + \underline{\varepsilon},\ \underline{\varepsilon} \sim N(\underline{0}, I)$, that the maximum likelihood estimate for $\underline{\beta}$ yields the ordinary least squares (OLS) answer. Suppose we assume a Laplacian prior on $\underline{\beta}$, i.e. $f\left(\underline{\beta}\right) \propto e^{-\lambda|\underline{\beta}|}$, the posterior distribution of $\underline{\beta}$ is given by

$$f\left(\underline{\beta} \mid \underline{y}\right) \propto f\left(\underline{\beta}\right).f\left(\underline{y} \mid \underline{\beta}\right) = e^{-\lambda\left|\underline{\beta}\right|}.\ e^{-\frac{1}{2}||\underline{y}-X\underline{\beta}||^2}.$$

This implies that the maximum a posteriori (MAP) estimate for $\underline{\beta}$ is given as

$$\underline{\hat{\beta}}_{MAP} = \arg\max f\left(\underline{\beta} \mid \underline{y}\right) = \arg\min ||\underline{y} - X\underline{\beta}||^2 + 2\lambda\left|\underline{\beta}\right|.$$

This is the same optimization as used in Lasso regression. Similarly, it is easy to see that assuming a Gaussian prior on $\underline{\beta}$, $f\left(\underline{\beta}\right) \propto .\ e^{-\frac{1}{2}||\underline{\beta}||^2}$ will coerce the MAP estimate to obtain the Ridge regression estimate. It is known from Bayesian analysis, that assuming a prior avoids overfitting by using our prior knowledge (in this case, knowledge of the parsimonious nature of the model); unsurprisingly, using a prior distribution and deriving the MAP estimate leads to robust regressors[32].

[31] For statistical – including Bayesian – inference, see Gelman et al (1995), Neal (1996). For Markov Chain Monte Carlo methods, see Gelfand and Smith (1990) and Spiegelhalter et al (1996). Expectation-Maximization (EM) algorithm is covered by Neal and Hinton (1998).

[32] The literature on Bayesian statistics is vast. Bayesian statistics for social sciences is covered in Hoff (2009), Gill (2002), Jackman (2009), Kruschke (2011), Christensen et al (2010). For reviewing application of Bayesian methods in different scenarios, see Breslow (1990) and the volumes by Gatsonis et al (1993-2002). The canonical result on exchangeability is from De Finetti (1974); see also Draper et al (1993). Historical development of Bayesian statistics is covered in Stigler (1986) – including the famous essays of Bayes (1763) and Laplace (1785, 1810). Non-informative priors are discussed in Jeffreys (1961), Kass and Wasserman (1996). Conjugate priors are covered in Box and Tiao (1973). Maximum entropy principle for constructing priors is discussed in Jaynes (1982, 1983), Donoho et al (1992). The debates between Bayesian and non-Bayesian approaches is given in Lindley (1958), Pratt (1965), Jaynes (1976), Berger and Wolpert (1984).

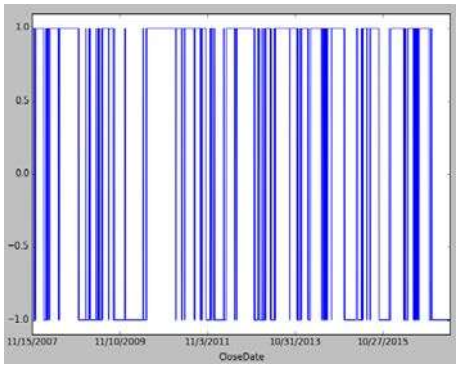
Example of Penalized Regression Approach in a Multi Asset Trend Following Strategy

We illustrate an application of Lasso by estimating the 1-day returns of assets in a cross-asset momentum model. For more background on trend following strategies see our CTA primer. We attempt to predict the returns of 4 assets: S&P 500, 7-10Y Treasury Bond Index, US dollar (DXY) and Gold. For predictor variables, we choose lagged 1M, 3M, 6M and 12M returns of these same 4 assets, yielding a total of 16 variables. To calibrate the model, we used a rolling window of 500 trading days (~2y); re-calibration was performed once every 3 months. The model was used to predict the next day's return. If the next day predicted return was positive, we went long the asset, otherwise we shorted it. Prior to regression, all inputs were standardized to avoid the problem of input features being of different scales. Performance of this momentum strategy is shown in tables below for each of the assets. Note that each of the momentum strategies outperformed a long only position in their respective asset.

S&P 500 (Lasso $\alpha = 0.001$) IR = 0.43

PnL vs Always-Long

Buy/Sell Indicator



Performance Analytics

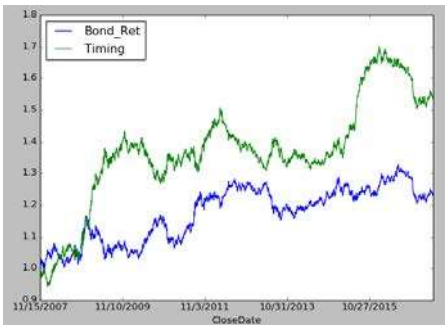
	Annualized Return (%)	Sharpe Ratio
S&P 500	7.52	0.36
S&P 500- Lasso	8.92	0.42
Correlation	41.3%	

Sample Signals Chosen (14 to 16 factors chosen)

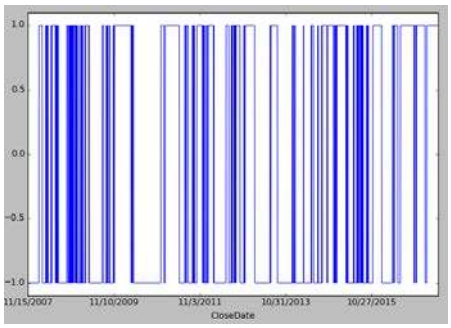
	1M	3M	6M	12M
Bond	✓	✓	✓	✓
Gold	✓	×	✓	✓
Dollar	×	✓	✓	✓
Equity	✓	✓	✓	✓

Result for IEF - Lasso ($\alpha = 0.001$) yields IR = 0.67

PnL vs Always-Long



Buy/Sell Indicator



Performance Analytics

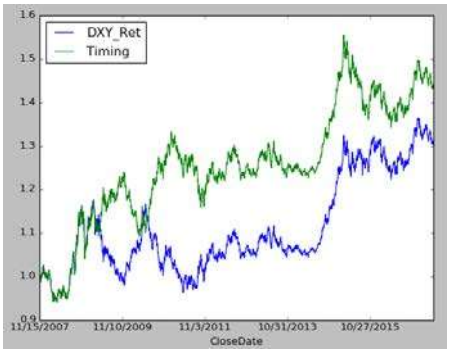
	Annualized Return (%)	Sharpe Ratio
IEF	2.30	0.32
IEF- Lasso	4.86	0.67
Correlation	-19.4%	

Sample Signals Chosen(12 to 16 factors chosen)

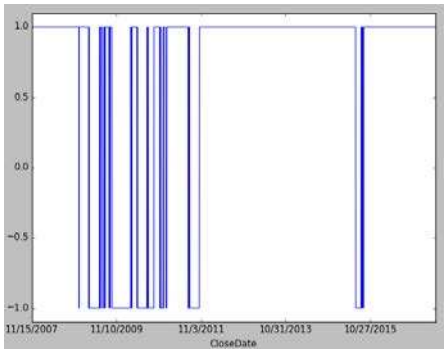
	1M	3M	6M	12M
Bond	✓	✓	✓	✓
Gold	✓	✓	✓	✓
Dollar	✓	✓	✓	✓
Equity	✓	×	✓	×

Result for DXY - Lasso (α = 0.05) yields IR = 0.50

PnL vs Always-Long



Buy/Sell Indicator



Performance Analytics

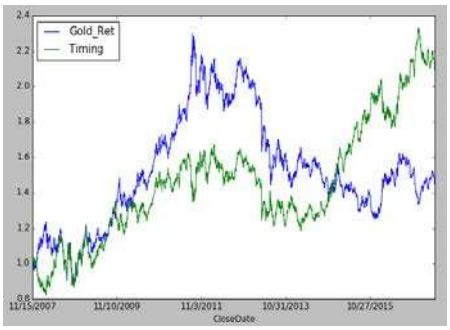
	Annualized Return (%)	Sharpe Ratio
DXY	3.22	0.38
DXY- Lasso	4.20	0.49
Correlation	58.9%	

Sample Signals Chosen(0 to 3 factors chosen)

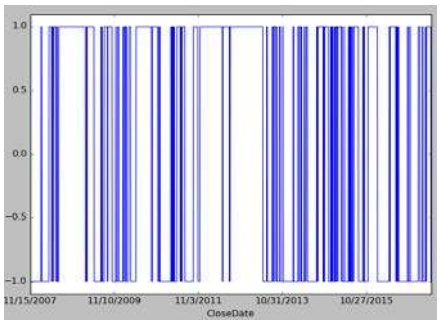
	1M	3M	6M	12M
Bond	×	×	×	×
Gold	×	×	✓	×
Dollar	×	×	×	×
Equity	✓	×	×	✓

Result for GLD - Lasso (α = 0.05) yields IR = 0.50

PnL vs Always-Long



Buy/Sell Indicator



Performance Analytics

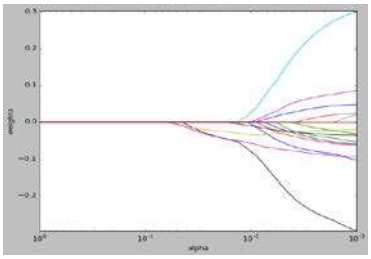
	Annualized Return (%)	Sharpe Ratio
Gold	6.12	0.31
GLD- Lasso	9.49	0.48
Correlation	20.3%	

Sample Signals Chosen(1 to 7 factors chosen)

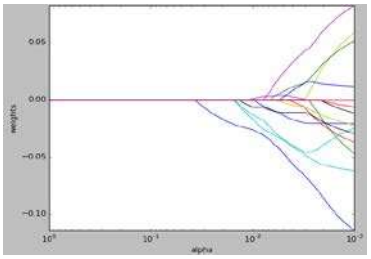
	1M	3M	6M	12M
Bond	×	✓	×	×
Gold	✓	×	×	✓
Dollar	✓	×	×	×
Equity	×	×	×	×

Evolution of betas as function of alpha in Lasso regression for 4 assets

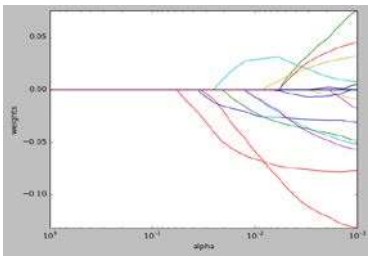
SPX



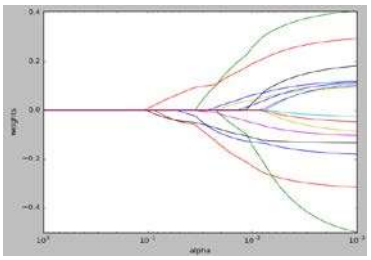
IEF



DXY



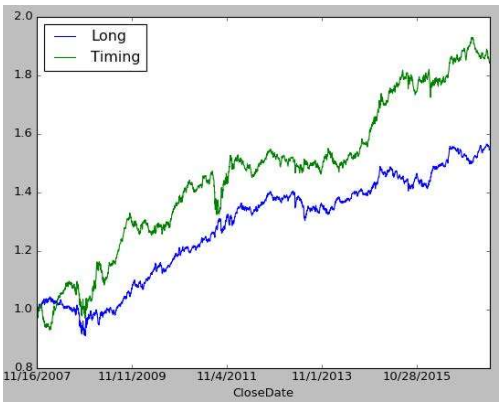
GLD



Combined Performance

Marko Kolanovic, PhD
(1-212) 272-1438
marko.kolanovic@jpmorgan.com

PnL vs Long : Cross-Asset



Performance Analytics

	Annualized Return (%)	Sharpe Ratio
X-A Long	4.80	0.75
X-A Lasso	6.80	0.85

Non-Parametric Regression: K-Nearest Neighbor and LOESS

Supervised learning algorithms are sometimes classified as being either parametric or non-parametric techniques. In parametric techniques, the model is described by a set of parameters such as linear regression beta (that is estimated from historical data). In non-parametric techniques, we do calibrate parameters of a model, but directly identify similar historical instances, and assume the behavior will be the same. Given a new datapoint, we search through historical data and identify a number 'K' of similar instances that we call "nearest neighbors" 'NN' (hence name K-NN). Then we then make predictions by averaging historical outcomes for these "nearest neighbors". Two examples of non-parametric regressions are.

- K-Nearest Neighbor (KNN)[33] rule: Once we have located K nearest neighbors, we can average the output variable y for this subset and use that as our prediction.

- LOESS: Using data for the K nearest neighbors, for each new point we fit a linear regression based on the K nearest neighbors (subset of historical data), and predict the output using those coefficients. This is called LOESS or localized linear regression.

Non-parametric techniques offer a simple way to extrapolate analysis on past similar events. KNN is commonly used by financial analysts, who intuitively employ it without referring to it as such. The use of a historical sample makes the technique "supervised", while the absence of parameters or coefficient betas makes it "non-parametric". In many financial analyses, the output variable is often not linearly related to the inputs; this makes linear regression and its extensions like ridge and lasso unsuitable. In such cases, the K-nearest neighbor can capture those non-linear properties. A drawback of using the KNN rule lies in its extreme sensitivity to outliers.

One can view linear regression and k-nearest neighbor methods as two ends of the spectrum of classical Machine Learning[34]. On one hand, linear regression 'under-fits' the data and hence suffers from higher 'bias' to achieve lower 'variance'. On the other hand, locating the K-most similar inputs and averaging their output makes a weak structural assumption. In formal terms, we say that the K-nearest neighbor method can 'over-fit' and hence suffer from higher 'variance' (while having a low 'bias').

We can gain further insight into the problem of under/over-fitting by varying the value of K in the KNN. At one extreme, we can set K to be equal to the number of samples itself, in which case we simply predict the quantized sample average as the output for all inputs. As we reduce K, we obtain a decision boundary that splits the training set into a number of parts. In the limiting case of K=1, we form a tiny region around each training sample. In this case, we have over-fit the data, which is likely to lead to a high error rate on unseen samples, while yielding a deceptively low (in this case, zero) error rate on the training set. One can informally think of a K-nearest neighbor method as dividing the N samples in historical data into N/K sets and computing their respective means. This implies that the effective number of parameters in K-nearest neighbor method is N/K, which increases as K decreases. The notion of effective number of parameters or degrees of freedom is a

[33] K-nearest neighbor rules and extensions are covered in Dasarathy (1991), Ripley (1996), MacQueen (1967), Hastie and Tibshirani (1996a), Simard et al (1993). For learning vector quantization, see Kohonen (1989). For discussion on Bayesian non-parametrics including Dirichlet Process Models, see Dunson (2009, 2010b), Shen and Ghosal (2013), Tokdar (2011), Wang and Dunson (2011a), Carvalho et al (2010), Ohlssen, Sharples and Spiegelhalter (2007), Ray and Mullick (2006), Rodriguez, Dunson and Gelfand (2009), Bigelow and Dunson (2009).

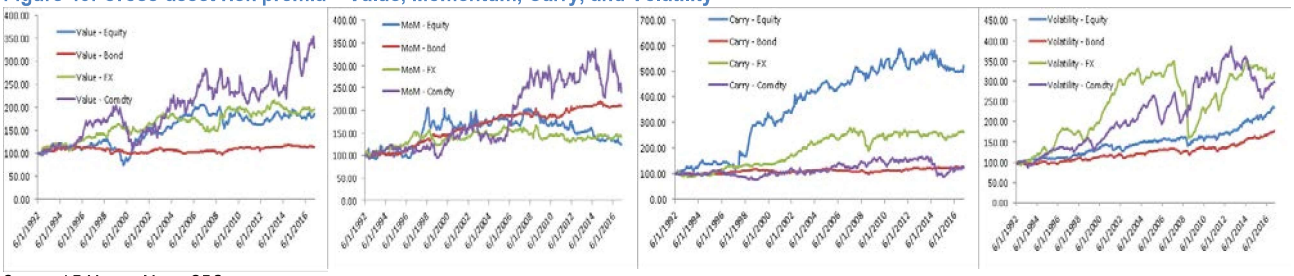
[34] Splines (including B-splines, thin-plate splines and reproducing kernel Hilbert spaces) are covered in Green and Silverman (1994), Wahba (1990), Girosi et al (1995) and Evgenion et al (2000). For wavelets, consult Daubechies (1992), Chni (1992), Wickerhauser (1994), Donoho and Johnstone (1994), Vidakovic (1999), Bruce and Gao (1996). Local regression and kernel methods are covered in Loader (1999), Fan and Gijbels (1996), Hastie and Tibshirani (1990). Statistical treatment of parametric non-linear models is illustrated in Reilly and Zeringue (2004), Gelman, Chew and Shnaidman (2004), Denison et al (2002), Chipman, George and McCulloch (1998, 2002), DiMatteo et al (2001) and Zhao (2000). For basis function models, see Bishop (2006), Biller (2000), DiMatteo, Genovese and Kass (2001), Barbieri and Berger (2004), Park and Casella (2008), Seeger (2008), Ramsay and Silverman (2005), Neelon and Dunson (2004), Dunson (2005), Hazelton and Turlach (2011), Hannah and Dunson (2011), Patti and Dunson (2011).

way to characterize the model complexity. As K decreases, and the number of parameters increases, the model complexity essentially increases, thereby leading to over-fitting.[35]

Financial Example

In our previous work, we studied cross-asset risk premia investing.[36] In particular, we had constructed simplified risk premia factors of value, momentum, carry, and volatility (across equities, bonds, currencies and commodities). Our analysis had shown that risk premia strategies delivered a positive Sharpe ratio over an extended period of ~40 years and exhibited low and stable correlation to traditional asset classes. We will attempt to use the K-nearest neighbor algorithm to illustrate potential for timing of these cross-asset risk premia based on macro regimes.

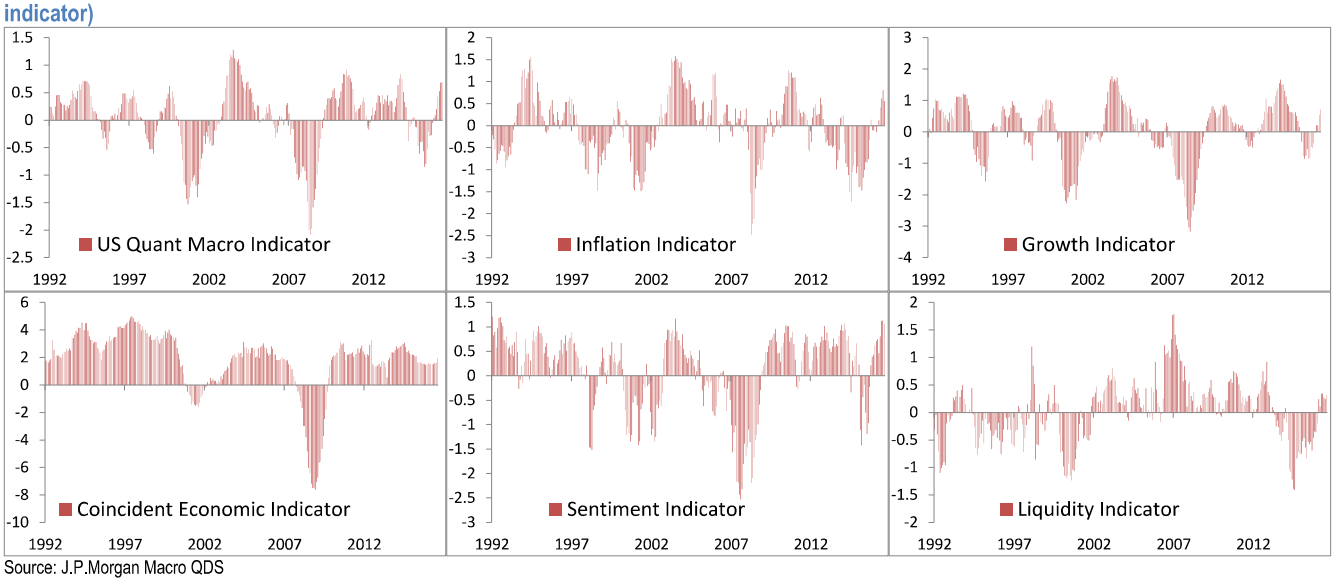
Figure 45: Cross-asset risk premia – Value, Momentum, Carry, and Volatility



Source: J.P.Morgan Macro QDS

To identify the current macro regime, we use 7 aggregated macro indicators: Quant Macro indicator, Growth, Liquidity, Sentiment, Inflation, Coincident economic indicator, as well as change in composite macro indicator. These aggregated indicators are based on 50 macro-economic signals; for details of construction, please see our earlier report titled "Framework for style investing: Style rotation and the business cycle".

Figure 46: Six aggregated macro indicators that are used to identify current economic regime (in addition, we use change in composite indicator)



Source: J.P.Morgan Macro QDS

[35] Formally, model complexity is characterized by Vapnik-Chervonenkis dimension of the set of classifiers or hypotheses. We describe that in a mathematical box in our discussion on model selection theory.

[36] Kolanovic, M and Wei, Z (2013), "Systematic strategies across asset classes: Risk factor approach to investing and portfolio management", J.P.Morgan Research Publication.

Figure 47: List of 50 macro-economic signals used to construct the 7 macro indicators above (for more details see here)

ISM Manufacturing PMI	ISM Non-manufacturing PMI
Initial Jobless Claims	US Capacity Utilization
US Leading Indicator	Yield Curve – 10y less 2y
Global Leading Indicator	Railroads Freight Index, 12m change
Leading-Lagging Economic Indicator Spread	US relative to World Stock Index
Baltic Dry Index	Global Economic Momentum
Manufacturing New Orders ex. Transportation	Retail Sales ex. Transportation
New Housing Permits	Credit Spread Moody's AAA-BAA
Michigan Consumer Sentiment	New Company Revisions, 3M Avg
NAPM Percentage Surprise, 3M Wt. Avg.	JULI Inv Grade USD Spread over Tsy
CESI – Citigroup Economic Surprise Index	VIX
Dow Transportation/Utilities, 12M Chg.	Small – Large Cap Outperformance, 12M Chg.
Barclays Investment Grade Spread	Barclays High Yield Spread
Term Structure of Momentum, Diffusion Index	Stock Market, 3Y Change
Margin Debt Level	Margin Debt as Percentage of Market Cap
Loan Officers Survey	M2 Money Stock, 12M Percentage Change
10Y Bond Yield, Real Rate	Correlation of MZM (Money) and SPX
Term Structure of Fed Fund Futures	USD Trade Weighted Index
Avg Treasury Trading Volume % of Mkt Debt	US Credit Manager Index
Free Liquidity	Earnings Yield Dispersion
Real Loan Growth	ISM Prices Index, 12M Change
PPI, YOY Percentage Change	Unit Labor Cost, 12M Change
US Import Price Inflation	Wage Trend Index, 12M Change
Breakeven Inflation. 10Y	Oil Price, WTI
Commodity Price Index	Median Home Price, 12M Change

Source: J.P.Morgan Macro QDS

We construct a monthly rebalancing strategy, where every month we look at a rolling window of the prior 10 years. We characterize the macro regime in a given month by the value of the 7 aggregated indicators. Then we look for K nearest neighbors, i.e.

a. We compute the distance between the economic regime vector of the current month to the economic regime vector for all months in the past 10 year window.

b. We rank them in ascending order of distance and choose the first (nearest) K neighbors

For each of these K nearest neighbors, we find the average return over a succeeding month for all 20 cross-asset risk premia factors. We then rank these factors according to their average returns, and choose a subset S that performed the best.

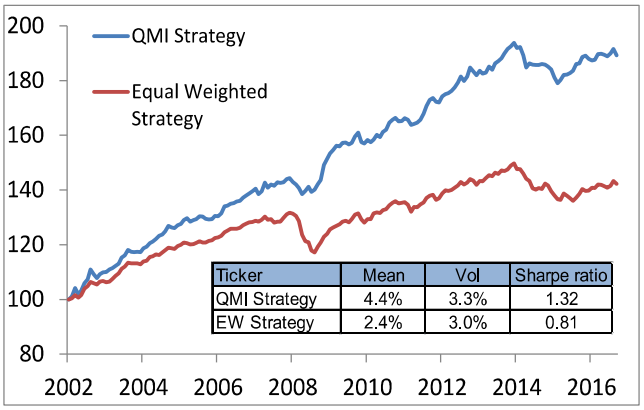
If we set S=1, it is equivalent to finding the best performing strategy historically in the same macro regime. If we set S=20, we will get an equal weighted strategy over the 20 risk premia. If we set K=1, we choose the nearest instance and replicate that for the current month. If we set K = 120, then we take all months in the input sample and average out the results, and invest in the best S strategies.

The Sharpe ratios of strategies based on K-nearest neighbors is tabulated below. We find that using K between 1 and 10, and number of risk factors between 10 and 19 generally outperforms simple equal weighted risk premia portfolio. For instance K=2 and S=14 yields the highest Sharpe ratio of 1.3, as compared to 0.8 shapre ratio of equal weighted strategy. We plot that case below alongside the equal weighted strategy that assigned equal weights to all 20 risk premia.

Marko Kolanovic, PhD
(1-212) 272-1438
marko.kolanovic@jpmorgan.com

Figure 48: Sharpe ratio of portfolio chosen using K-Nearest Neighbor rule over cross-asset systematic strategies (left, K=1 to 25); Performance by timing cross-asset risk premia using JPM's Quant Macro Indicators (right)

	Number of Risk Premia Strategy																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.6	0.6	0.7	0.8	0.8	0.9	1.0	1.1	1.2	1.2	1.2	1.3	1.3	1.3	1.3	1.3	1.2	1.1	0.9	0.8
2	0.4	0.6	0.7	0.8	0.8	0.9	0.9	1.0	1.1	1.1	1.1	1.2	1.2	1.3	1.3	1.3	1.2	1.2	1.0	0.8
3	0.1	0.3	0.6	0.7	0.8	0.8	0.9	0.9	0.9	0.9	1.0	1.0	1.1	1.1	1.1	1.1	1.1	1.1	1.0	0.8
4	0.2	0.2	0.3	0.5	0.6	0.6	0.7	0.7	0.8	0.8	0.8	0.8	0.9	0.8	0.9	1.0	0.9	1.0	1.0	0.8
5	0.0	0.3	0.4	0.5	0.5	0.7	0.7	0.8	0.8	0.8	0.8	0.9	1.0	1.0	1.1	1.1	1.0	1.0	0.9	0.8
6	0.2	0.3	0.4	0.3	0.3	0.3	0.3	0.6	0.6	0.6	0.8	0.9	0.9	0.9	0.9	0.9	1.0	1.0	1.0	0.8
7	-0.1	0.3	0.4	0.4	0.4	0.3	0.4	0.5	0.5	0.6	0.7	0.8	0.9	0.9	0.9	1.0	1.0	0.9	0.9	0.8
8	0.2	0.2	0.2	0.4	0.6	0.5	0.6	0.7	0.7	0.8	0.8	0.9	1.0	0.9	0.9	1.0	1.0	1.0	0.9	0.8
9	0.1	0.2	0.2	0.5	0.6	0.7	0.7	0.7	0.8	0.9	1.0	0.9	1.0	1.0	1.0	0.9	0.9	0.9	0.9	0.8
10	0.0	0.1	0.2	0.3	0.5	0.5	0.6	0.7	0.7	0.8	0.7	0.8	0.8	0.9	0.8	0.9	0.9	0.9	0.9	0.8
11	-0.1	0.2	0.3	0.3	0.4	0.4	0.5	0.5	0.6	0.7	0.8	0.8	0.8	0.9	0.9	0.9	0.9	0.9	0.8	0.8
12	0.1	0.0	0.1	0.3	0.5	0.4	0.5	0.6	0.7	0.7	0.8	0.9	0.9	0.9	0.9	1.0	0.9	0.9	0.8	0.8
13	0.0	0.1	0.2	0.3	0.4	0.6	0.7	0.7	0.7	0.7	0.7	0.7	0.9	0.9	1.0	1.0	1.0	1.0	0.9	0.8
14	0.1	0.3	0.4	0.3	0.4	0.5	0.6	0.7	0.7	0.8	0.7	0.7	0.8	0.8	0.9	0.9	0.9	1.0	0.9	0.8
15	0.0	0.3	0.3	0.4	0.5	0.7	0.7	0.6	0.6	0.6	0.7	0.8	0.9	1.0	1.0	1.0	0.9	0.9	0.9	0.8
16	-0.1	0.1	0.3	0.4	0.6	0.6	0.7	0.7	0.7	0.7	0.7	0.7	0.8	0.9	0.9	0.9	0.9	0.9	0.9	0.8
17	0.1	0.2	0.2	0.5	0.3	0.4	0.6	0.6	0.6	0.7	0.7	0.8	0.8	0.9	1.0	0.9	0.9	0.9	0.9	0.8
18	0.2	0.4	0.3	0.5	0.6	0.5	0.6	0.7	0.7	0.8	0.8	0.8	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.8
19	0.2	0.2	0.1	0.4	0.5	0.6	0.7	0.7	0.7	0.6	0.7	0.8	0.9	0.8	0.8	0.9	0.9	0.9	0.9	0.8
20	0.3	0.1	0.3	0.4	0.6	0.6	0.6	0.7	0.7	0.7	0.7	0.8	0.9	0.9	0.8	0.8	0.9	0.9	1.0	0.8
21	0.1	0.2	0.4	0.6	0.7	0.6	0.6	0.6	0.7	0.7	0.7	0.7	0.7	0.8	0.8	0.8	0.9	0.9	0.9	0.8
22	0.3	0.1	0.3	0.5	0.6	0.5	0.5	0.5	0.6	0.6	0.6	0.7	0.7	0.7	0.8	0.8	0.9	0.8	0.8	0.8
23	0.3	0.4	0.3	0.6	0.7	0.5	0.6	0.6	0.7	0.7	0.7	0.7	0.8	0.7	0.8	0.8	0.8	0.9	0.9	0.8
24	-0.1	0.3	0.3	0.4	0.6	0.5	0.6	0.6	0.7	0.8	0.7	0.7	0.6	0.7	0.8	0.8	0.8	0.8	0.9	0.8
25	-0.1	0.3	0.4	0.5	0.6	0.6	0.6	0.7	0.7	0.7	0.6	0.7	0.7	0.7	0.8	0.8	0.8	0.8	0.8	0.8



Source: J.P.Morgan Macro QDS