ML3 – dimensionality and assessment

Machine Learning – Tools and applications for policy – Lecture 4

Iman van Lelyveld – Michiel Nijhuis

DNB Data Science Hub

ML3 – dimensionality and assessment

1. How to reduce dimensionality?
   - Principal Components Analysis (PCA)
2. Feature selection and regularization
   - How to tune model input by selecting features and beat overfitting?
   - How to select the most important features?
   - Examples RIDGE, LASSO, Elastic net
3. Is a "good" model always good? What is external validity?
   - Holdout, K-fold cross validation, Stratified K-fold. Leave-one-out (LOO)
4. What if we apply this to asset pricing?

Outline

ML3 – dimensionality and assessment
- Dimensionality Reduction
  - Principal Component Analysis (PCA)
- Feature selection
  - Further discussion of Regularization
  - Least Absolute Shrinkage and Selection Operator (LASSO)
- Model Evaluation and Hyperparameter Tuning
- Information leakage
- Taming the factor zoo

Knowledge clips

- Principal component analysis – PCA (Statquest) (link)
  The key concepts are covered until 12.35.
- RIDGE regressions (Statquest) (link)
  This video is a bit slow but does cover RIDGE in detail
- LASSO regressions (Statquest) (link)
  If you have just seen RIDGE regression, you can start at 2.40
- Bonus: Playing around with Eigenvectors (Victor Powell and Lewis Lehe)

Outline

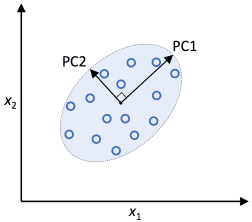
ML3 – dimensionality and assessment
- Dimensionality Reduction
  - Principal Component Analysis (PCA)
- Feature selection
  - Further discussion of Regularization
  - Least Absolute Shrinkage and Selection Operator (LASSO)
- Model Evaluation and Hyperparameter Tuning
- Information leakage
- Taming the factor zoo

Data and feature selection

- Economic theory and practitioner knowledge is usually the starting point for looking for a parsimonious model
- Examining the correlation matrix of the features included can provide insight in (non-linear) relations/dependencies
- However, if the # of features increases, statistical methods to reduce dimensionality are called for: Principal Component Analysis (PCA)
- Also, we can aim to reduce the number of features
  - Least Absolute Shrinkage and Selection Operator (LASSO)
  - Least Angle Regression (LARS), RIDGE (Hastie et al. (2017))
- Efficient feature selection can also save computational and memory resources, in addition to improving model performance.

Principal Component Analysis (PCA)

- Find the directions of maximum variance
  - Transforming/Projecting d-dimensional data to k dimensions ($k << d$)
- Principal components: PC1 and PC2
  - First principal component will have the largest variance
  - Second principal component will have next largest variance, etc., ...
- PCA sensitive to scaling, so need to standardize features



See "Principal component analysis" – PCA (Statquest) in Knowledge clips.

A roadmap for PCA

Steps:

1. Standardize the d-dimensional dataset
2. Construct the covariance matrix
3. Decompose the covariance matrix into its eigenvectors and eigenvalues
4. Select k eigenvectors that correspond to the k largest eigenvalues, where k is the dimensionality of the new feature subspace ($k \leq d$).
5. Construct a projection matrix $\mathbf{W}$ from the "top" k eigenvectors
6. Transform the d-dimensional input dataset $\mathbf{X}$ using the projection matrix $\mathbf{W}$ to obtain the new k-dimensional feature subspace
7. See Statquest link for PCA Step-by-Step explanation.

Step 2 Variance-covariance matrix

- Symmetric $d \times d$ -dimensional matrix (d - number of dimensions)
- Pairwise covariances between the different features
- Covariance between two features $\mathbf{x}_j$ and $\mathbf{x}_k$:

$$cov(x_j, x_k) = \frac{1}{n} \sum_{i=1}^{n} \left(x_j^{(i)} - \mu_j\right)\left(x_k^{(i)} - \mu_k\right)$$

Where μ_j and μ_k are the sample means of feature j and k (i.e., expected values)

- Covariance can be standardized to yield the correlation

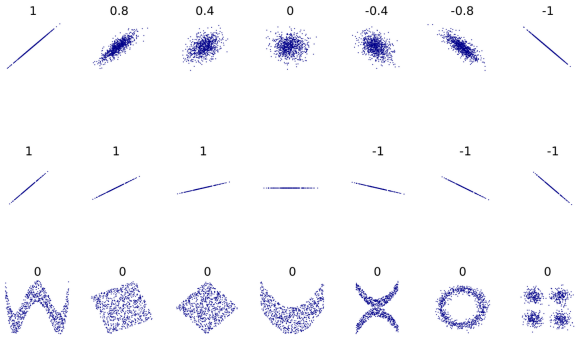
$$\rho_{j,k} = \frac{cov(x_j, x_k)}{\sigma_j \sigma_k}$$

What is covariance?

- Measure of how much two random variables change together
- Positive covariance
  - Features increase and decrease together
  - e.g. as a balloon is blown up it gets larger in all dimensions
- Negative covariance
  - Features vary in opposite directions
  - Large values of one variable correspond to small values of the other
  - e.g. if a balloon is squashed in one dimension then it will expand in the other two
- The magnitude of the covariance is not easy to interpret
- PM: The normalized version of covariance (correlation coefficient) indicates the strength of the linear relation

Linear dependence – Correlation

- correlation indicates the degree of the linear co-movement between two variables ...
- ... but does not say anything about the informativeness
- Nor anything about non-linearity



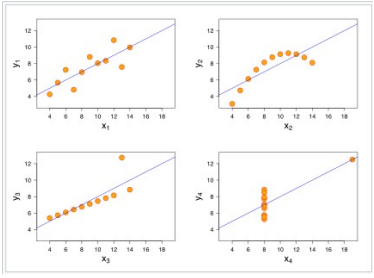
Anscombe (1973)

Outliers matter

Anscombe's quartet (Anscombe (1973))

Clockwise from top left:

- simple linear relationship
- not linear and the Pearson correlation coefficient is not relevant. Regression with non-linear features
- one high-leverage point is enough to produce a high correlation coefficient even without a relationship between the variables
- linear but with a different regression line: adding the outlier changes the slope coefficient from 1 to 0.816



Step 3 Eigenvectors and eigenvalues

- For two features, covariance matrix will look like this:

$$A = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix}$$

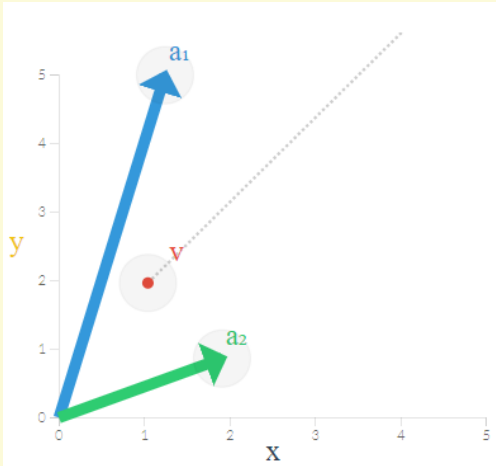
- The eigenvector of A represent the principal components: direction of maximum variance
- The corresponding eigenvalues represent their magnitude
- More formally: An Eigenvector $\mathbf{v}$ satisfies the condition:

$$A\mathbf{v} = \lambda\mathbf{v}$$

where λ is the eigenvalue (scalar)

See "Playing around with Eigenvectors" (Powell and Lehe) in Knowledge clips

Interacting with eigenvectors and eigenvalues



$$A = \begin{bmatrix} a_{1,x} & a_{2,x} \\ a_{1,y} & a_{2,y} \end{bmatrix} = \begin{bmatrix} 1.24 & 1.91 \\ 5.00 & 0.86 \end{bmatrix}$$

$$v = \begin{bmatrix} 1.04 \\ 1.97 \end{bmatrix}$$

$$Av = \begin{bmatrix} 5.04 \\ 6.90 \end{bmatrix}$$

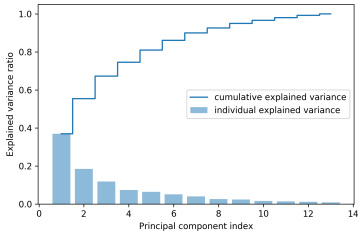
Source: Setosa.io

Variance explained ratio

- Variance explained ratio of an eigenvalue λ_j:

$$\frac{\lambda_j}{\sum_{j=1}^{d} \lambda_j}$$

- First two principal components explain about 60 percent of the variance in the data
- Choosing the 'optimal' number of PCs: elbow? ad hoc?



Outline

ML3 – dimensionality and assessment
- Dimensionality Reduction
  - Principal Component Analysis (PCA)
- Feature selection
  - Further discussion of Regularization
  - Least Absolute Shrinkage and Selection Operator (LASSO)
- Model Evaluation and Hyperparameter Tuning
- Information leakage
- Taming the factor zoo

$L1$ regularization

Recall from our discussion on Logit, $L2$ regularization – one approach to reduce model complexity

$$L2 : \|\mathbf{w}\|_2^2 = \sum_{j=1}^{m} w_j^2$$

An alternative approach is $L1$ regularization:

$$L1 : \|\mathbf{w}\|_1 = \sum_{j=1}^{m} |w_j|$$

$L1$ regularization

- $L1$ yields sparse solutions
- Most feature weights will be zero
- Useful for high-dimensional datasets with irrelevant features
- It can be viewed as a technique for feature selection
- Some intuition as to why this is the case will follow. For OLS, the cost function becomes:

$$J(\mathbf{w}) = \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2 + \lambda \|\mathbf{w}\|$$

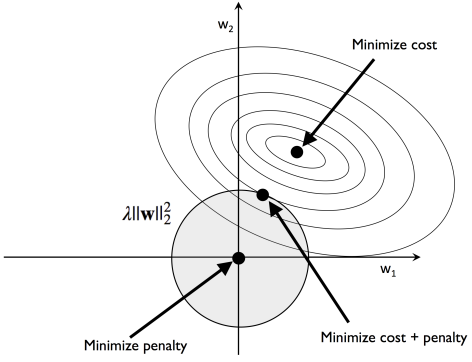
which becomes a regular OLS with $\lambda \to 0$

Outline

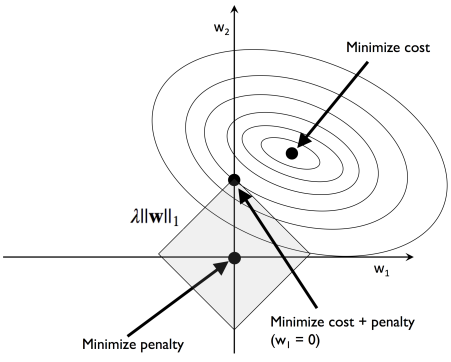
ML3 – dimensionality and assessment
- Dimensionality Reduction
  - Principal Component Analysis (PCA)
- Feature selection
  - Further discussion of Regularization
  - Least Absolute Shrinkage and Selection Operator (LASSO)
- Model Evaluation and Hyperparameter Tuning
- Information leakage
- Taming the factor zoo

Applying regularization – Lasso

L2: RIDGE



L1: Lasso



See both Statquest clips in Knowledge clips

Sparsity

- Regularization penalty and cost pull in opposite directions
- Regularization wants the weight to be at (0, 0)
  - i.e. regularization prefers a simpler model
  - note that Lasso can have a zero weight on a feature (because the 'triangle' has corners where one of the weights is zero)
- Decreases the dependence of the model on the training data

*L*1 in scikit-learn

Balancing Ridge and Lasso: Elastic Net

- Limitation of Ridge Regression
  - Ridge regression decreases the complexity of a model but does not reduce the number of variables since it never leads to a coefficient been zero rather only minimizes it. Hence, this model is not good for feature reduction.
- Limitation of Lasso Regression
  - If there are two or more highly collinear variables then LASSO regression select one of them randomly which is not good for the interpretation of data
  - Lasso sometimes struggles with some types of data. If the number of predictors (p) is greater than the number of observations (n), Lasso will pick at most n predictors as non-zero, even if all predictors are relevant (or may be used in the test set).

Solution: weigh Ridge and Lasso $\rightarrow$ Elastic Net

Outline

ML3 – dimensionality and assessment
- Dimensionality Reduction
  - Principal Component Analysis (PCA)
- Feature selection
  - Further discussion of Regularization
  - Least Absolute Shrinkage and Selection Operator (LASSO)
- Model Evaluation and Hyperparameter Tuning
- Information leakage
- Taming the factor zoo

How do we know the model is working?

- Model evaluation
  - Performance metrics (discussed in Lecture 3 - ML2 – the basics) indicate how good the model is
- How do we obtain an unbiased estimate of model's performance?
- Key concept: estimate model performance on unseen data
- So we need to separate data for 1) finding the right model and for 2) assessing it
- Note the difference between model parameters (e.g. weights) vs hyperparameters (e.g. de *k* in *k*-nearest neighbors)

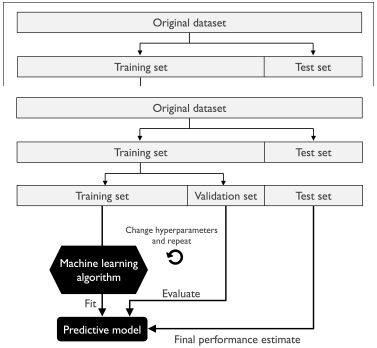
There are numerous ways to check validity

1. Holdout
2. *k*-fold
3. Stratified *k*-fold
4. Leave-one-out
5. …

See SKLearn documentation for further approaches

1. The holdout method

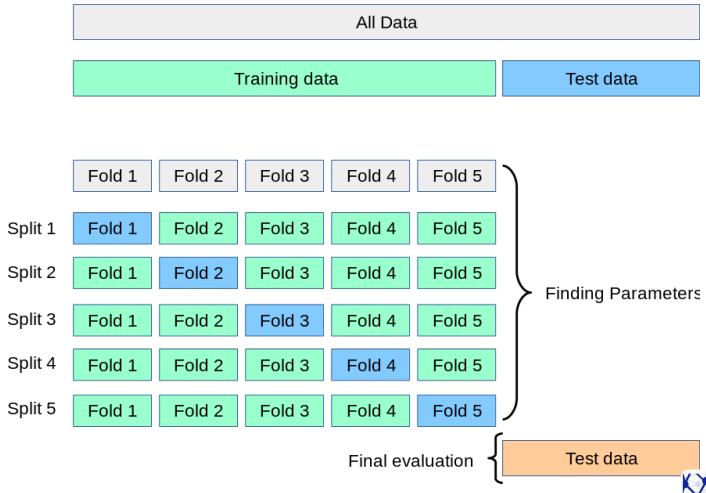
- Split data into training and test datasets
- However, typically we cannot test immediately after training
  - Need to tune the model to further improve the performance
  - Select optimal values of hyperparameters
- This step is known as model selection
- A better approach: training + validation + test sets
  - Validation set is used for model selection
  - Test set is for model evaluation
- Let's for now assume we know the hyperparameters



2. *K*-fold cross-validation

- Disadvantage of the holdout method: sensitive to partitioning
- To fix this, use *K*-fold cross-validation
  - Randomly split the training dataset into k folds
  - Of these, $k - 1$ folds are used for training and one for testing
  - Repeat this procedure k times and then average across k folds
- Each sample will be part of train and test sets
- Lower-variance estimate of the model performance (compared to holdout)

K-fold cross-validation

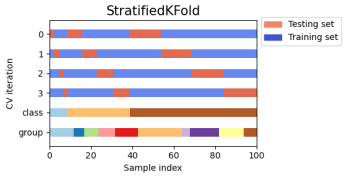


How do we pick the number of folds?

- The standard value is $k = 10$
- For small datasets, increase the number of folds
  - increases the amount of training data
- For larger datasets, we can decrease the number of folds
  - e.g. $k = 5$ is a reasonable choice

3. Stratified *k*-fold cross-validation

- *StratifiedKFold* is a variation of k-fold which returns stratified folds: Class proportions preserved in each fold
- So each fold is representative of the entire training set
- Better performance estimates for imbalanced data

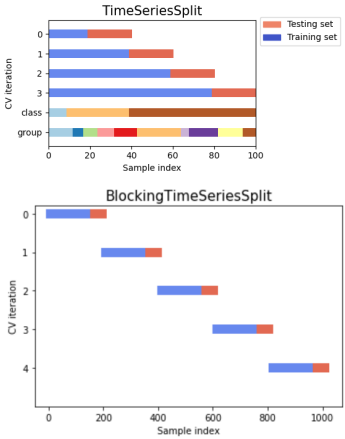


4. Leave-one-out cross-validation

- Set the number of folds equal to the number of training samples
- Only a single training sample used for testing during each iteration
- Recommended approach for very small datasets

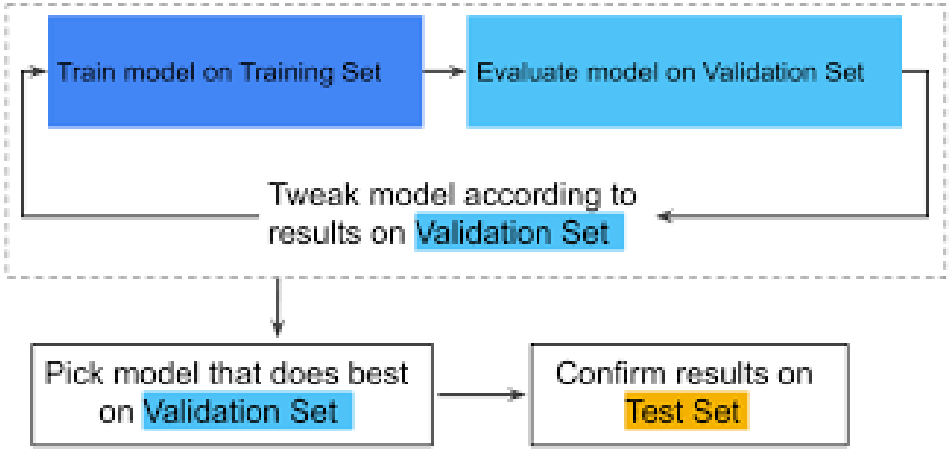
Time series make things even more complicated

- *TimeSeriesSplit* is a variation of *k*-fold which returns first *k* folds as train set and the (*k* + 1) th fold as test set. Note that unlike standard cross-validation methods, successive training sets are supersets of those that come before them. Also, it adds all surplus data to the first training partition, which is always used to train the model.
- Alternatively you can do walk forward cross validation



DataScience Hub

How to find the right hyperparameters



How to find the right hyperparameters

- Many ML algorithms offer a number of hyperparameters, but it is often unclear what the optimal set is
- Options
  1. Manual
  2. Grid search: a brute-force exhaustive search of the complete hyperparameter space
     - `GridSearchCV`
     - Obviously, this can be computationally very expensive but will find the global optimum
  3. Randomized search:
     - `RandomizedSearchCV`

Overview of hyper parameters in Python packages

ML Algorithm	Main HPs	Optional HPs	HPO methods	Libraries
Linear regression	–	–	–	–
Ridge & lasso	alpha	–	BO-GP	Skpot
Logistic regression	penalty, c, solver	-	BO-TPE, SMAC	Hyperopt, SMAC
KNN	n_neighbors	weights, p, algorithm	BOs, Hyperband	Skpot, Hyperopt, SMAC, Hyperband
SVM	C, kernel, epsilon (for SVR)	gamma, coef0, degree	BO-TPE, SMAC, BOHB	Hyperopt, SMAC, BOHB
NB	alpha	–	BO-GP	Skpot
DT	criterion, max_depth, min_samples_split, min_samples_leaf, max_features	splitter, min_weight_fraction_leaf, max_leaf_nodes	GA, PSO, BO-TPE, SMAC, BOHB	TPOT, Optunity, SMAC, BOHB
RF & ET	n_estimators max_depth, criterion, min_samples_split, min_samples_leaf, max_features	splitter, min_weight_fraction_leaf, max_leaf_nodes	GA, PSO, BO-TPE, SMAC, BOHB	TPOT, Optunity, SMAC, BOHB
XGBoost	n_estimators, max_depth, learning_rate, subsample, colsample_bytree,	min_child_weight, gamma, alpha, lambda	GA, PSO, BO-TPE, SMAC, BOHB	TPOT, Optunity, SMAC, BOHB
Voting	estimators, voting	weights	GS	Sklearn
Bagging	base_estimator, n_estimators	max_samples, max_features	GS, BOs	sklearn, Skpot, Hyperopt, SMAC
AdaBoost	base_estimator, n_estimators, learning_rate	-	BO-TPE, SMAC	Hyperopt, SMAC
Deep learning	number of hidden layers, 'units' per layer, loss, optimizer, Activation, learning_rate, dropout rate, epochs, batch_size, early stop patience	number of frozen layers (if transfer learning is used)	PSO, BOHB	Optunity, BOHB
K-means	n_clusters	init, n_init, max_iter	BOs, Hyperband	Skpot, Hyperopt, SMAC, Hyperband
Hierarchical clustering	n_clusters, distance_threshold	linkage	BOs, Hyperband	Skpot, Hyperopt, SMAC, Hyperband
DBSCAN	eps, min_samples	-	BO-TPE, SMAC, BOHB	Hyperopt, SMAC, BOHB
Gaussian mixture	n_components	covariance_type, max_iter, tol	BO-GP	Skpot
PCA	n_components	svd_solver	BOs, Hyperband	Skpot, Hyperopt, SMAC, Hyperband
LDA	n_components	solver, shrinkage	BOs, Hyperband	Skpot, Hyperopt, SMAC, Hyperband

Source: Yang and Shami 2020

Comparison of Hyperparameter Optimization Algorithms

HPO Method	Strengths	Limitations	Time Complexity
GS	• Simple.	• Time-consuming • Only efficient with categorical HPs.	$O(n^k)$
RS	• More efficient than GS • Enable parallelization.	• Not consider previous results • Not efficient with conditional HPs.	$O(n)$
Gradient-based models	• Fast convergence speed for continuous HPs.	• Only support continuous HPs • May only detect local optimums.	$O(n^k)$
BO-GP	• Fast convergence speed for continuous HPs.	• Poor capacity for parallelization • Not efficient with conditional HPs.	$O(n^3)$
SMAC	• Efficient with all types of HPs.	• Poor capacity for parallelization.	$O(nlogn)$
BO-TPE	• Efficient with all types of HPs • Keep conditional dependencies.	• Poor capacity for parallelization.	$O(nlogn)$
Hyperband	• Enable parallelization.	• Not efficient with conditional HPs • Require subsets with small budgets to be representative.	$O(nlogn)$
BOHB	• Efficient with all types of HPs • Enable parallelization.	• Require subsets with small budgets to be representative.	$O(nlogn)$
GA	• Efficient with all types of HPs • Not require good initialization.	• Poor capacity for parallelization.	$O(n^2)$
PSO	• Efficient with all types of HPs • Enable parallelization.	• Require proper initialization.	$O(nlogn)$

Algorithm	Advantages	Disadvantages
Grid search	Simple	Time-consuming. Only efficient with

Hyperparameters: Gradient Boosting Decision Trees

					xgboost			lightgbm			catboost	
					Parameter Names			Parameter Names			Parameter Names	
Name	What	Allowed Range	Baseline Choice	Favorable Tuning Range	Original API	sklearn API	Default	Original API	sklearn API	Default	Original API & sklearn API	Default
1. Most important!												
Maximum Depth	Maximum depth of each trained tree.	[0, ∞]	5 or 6	[3, 12]	max_depth	max_depth	6	max_depth	max_depth	-1	depth	6
2. Tune at earlier phase.												
Row Sampling	Percentage of rows used per iteration frequency.	(0, 1]	0.8	[0.6, 1.0]	subsample	subsample	1.0	bagging_fraction	subsample	1.0	subsample	Depends
Column Sampling by Tree	Percentage of columns used per iteration.	(0, 1]	0.8	[0.6, 1.0]	colsample_bytree	colsample_bytree	1.0	feature_fraction	colsample_bytree	1.0	NULL	
Column Sampling by Level	Percentage of columns used per split selection.	(0, 1]	0.5	[0.6, 1.0]	colsample_bylevel	colsample_bylevel	1.0	NULL			rsm	1.0
Hessian Regularization	Prune by minimum hessian requirement.	[0, ∞]	1.0	[0.1, 10.0]	min_child_weight	min_child_weight	1.0	min_sum_hessian_in_leaf	min_child_weight	0.001	NULL	
3. Tune at intense tuning phase.												
Minimum Data per Leaf	Prune by minimum number of observations requirement.	[0, ∞]	Depends on data size. 10 or 20 for small data (#<100), hundreds/thousands for large data.		NULL			min_data_in_leaf	min_child_samples	20	min_data_in_leaf *	1
Maximum Leaves	Maximum leaves for each trained tree.	[1, ∞]	255	xgboost, catboost: 255/ [2^6] ~ 255* [2^6] lightgbm: Less than 2^(max_depth)	max_leaves	max_leaf_nodes	0	num_leaves	num_leaves	31	max_leaves *	31
4. Parameters to tune when other parameters are tuned enough.												
L1 Regularization	L1 Regularization for boosting.	[0, ∞]	Default Value		alpha	reg_alpha	0.0	lambda_l1	reg_alpha	0.0	NULL	
L2 Regularization	L2 Regularization for boosting.	[0, ∞]	Default Value		lambda	reg_lambda	1.0	lambda_l2	reg_lambda	0.0	l2_leaf_reg	3.0
Loss Regularization	Prune by minimum loss requirement.	[0, ∞]	Default Value	[0.0, 1.0]	gamma	gamma	0.0	min_gain_to_split	min_split_gain	0.0	NULL	
5. Leave baseline at first, smaller at the time of final tuning.												
Learning rate	Multiplication performed on each boosting iteration.	(0, 1]	0.1	[0.01, 0.05] as phase advances.	eta	learning_rate	0.3	learning_rate	learning_rate	0.1	learning_rate	Auto or 0.03
6. Too many iterations can cause overfitting. Number of iterations can be large if early stopping is enabled, and we should do so :)												
Number of Iterations	Number of boosting iterations.	[1, ∞]	1000 or 10000 with early stopping.		nrounds	n_estimators	NULL	num_iterations	n_estimators	100	iterations	1000
Early Stopping	Number of maximum iterations without improvements.	[0, ∞]	10 divided by 'learning rate'		early_stopping_rounds	early_stopping_rounds	NULL	early_stopping_round	early_stopping_rounds	0	early_stopping_rounds	False

* Some catboost parameters are only available in traing on GPU.

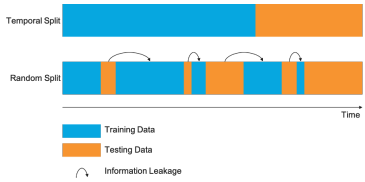
DataScience Hub

Outline

ML3 – dimensionality and assessment
- Dimensionality Reduction
  - Principal Component Analysis (PCA)
- Feature selection
  - Further discussion of Regularization
  - Least Absolute Shrinkage and Selection Operator (LASSO)
- Model Evaluation and Hyperparameter Tuning
- Information leakage
- Taming the factor zoo

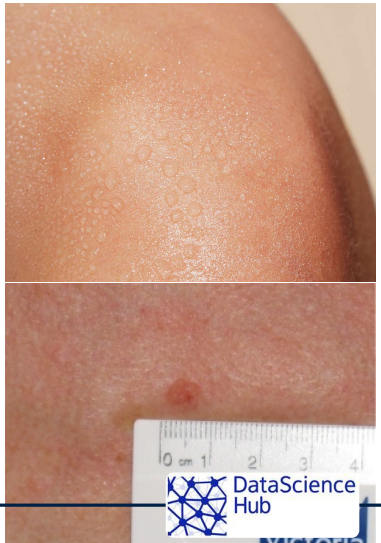
Information leakage

- Explicit: some features represent transformation of or a proxy for the target variable
  - Number of months behind in interest payments
- Implicit: training has info unavailable for the test observations
  - time series: if a feature is measured with a lag and/or is revised often (e.g. GDP, news), be careful to use the right value



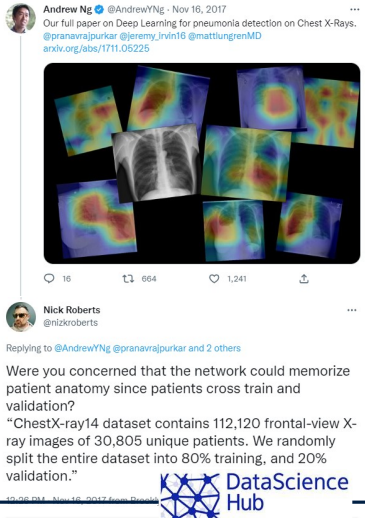
Use case: rulers cause skin cancer

- Skin cancer, the most common human malignancy, is primarily diagnosed visually, beginning with an initial clinical screening and followed potentially by more (invasive) tests
- Esteva et al. (2017, Nature), train a CNN using a dataset of 129,450 clinical images consisting of 2,032 different diseases. We test its performance against 21 board-certified dermatologists on biopsy-proven clinical images
- It would be great if this method could be rolled out to your smartphone ...
- ... but turns out that a ruler is bad for you



Use case: the random shuffle

- CheXNet: algorithm to detect pneumonia from chest X-rays at a level exceeding radiologists (Rajpurkar et al. (2017))
- 121-layer convolutional neural network trained on largest chest X-ray dataset (ChestX-ray14)
- Radiologists annotate a test set (n=4), on which we compare CheXNet to radiologists performance
- CheXNet > average radiologist performance on the F1 metric
  - 100,000 frontal-view X-rays with 14 diseases
  - 30,000 patients
  - random train-test



DataScience Hub

Outline

ML3 – dimensionality and assessment
- Dimensionality Reduction
  - Principal Component Analysis (PCA)
- Feature selection
  - Further discussion of Regularization
  - Least Absolute Shrinkage and Selection Operator (LASSO)
- Model Evaluation and Hyperparameter Tuning
- Information leakage
- Taming the factor zoo

Applications

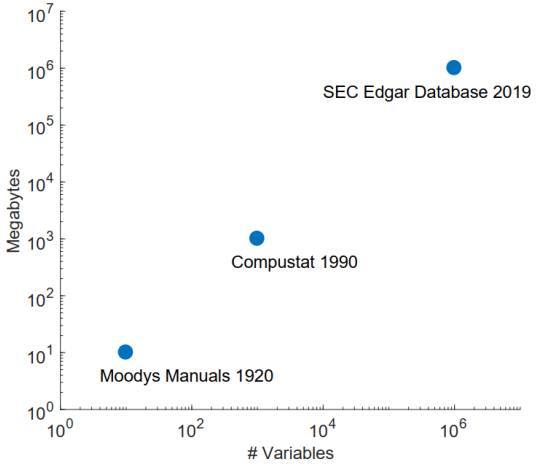
- So far we've talked a lot about techniques and relatively little about applications for finance such as:
  - Classification: robo advice, fraud detection
  - Forecasting: trading bots
  - NLP: compliance
- Here we will look at one example: asset pricing based on Kozak et al. (2019)
- Also see Bianchi, Büchner, Hoogteijling, and Tamoni (2021), Bianchi, Büchner, and Tamoni (2021), Chen (2021), Easley et al. (2021), Erel et al. (2021), Farboodi et al. (2022), Fuster et al. (2022), Goldstein et al. (2021), Leippold et al. (2022), Li et al. (2021), and Obaid and Pukthuanthong (2022)

How would all of this work in asset pricing?

- Prediction is central to ML and also essential to asset pricing (AP)
  - Forecasting returns
  - Forecasting cash-flows
  - Forecasting default
  - Forecasting risk exposures
- Fundamental asset pricing equation for asset with excess return R and Stochastic Discount Factor (SDF) M:
  $\mathbb{E}[R_{t+1}M_{t+1}|x_t] = 0$
- Empirical implementation involves function approximation $x_t \rightarrow$ (Co-)moments of $R_{t+1}; M_{t+1}$
- This is a supervised learning problem + maybe dimension reduction in joint distribution of $(R_{t+1}; M_{t+1})$: unsupervised learning
- Pre-ML literature: x_t typically low-dimensional but little real-world justification

This section is based on Kozak et al. (2019)

More data available to extend the set of x_t in asset pricing



Source: Nagel (2019)

DataScience Hub

Overfitting! – bring in prior knowledge through regularization

- Consider supervised learning problem: find $y_i = f(x_i)$ where $i = 1, 2, \ldots, N$ and x_i has dimension $J \times 1$.
- When x_i high-dimensional (e.g., $J > N$), standard methods (e.g., OLS) would horribly overfit in-sample $\rightarrow$ bad out-of-sample (OOS) prediction performance
- Regularization: Penalize estimation results that are regarded as implausible based on prior knowledge
  - Example: if big magnitudes of regression coefficient on Sharpe ratio are a priori unlikely, penalize big coefficient estimates
- Remember: many ML methods can be derived as penalized estimators

Many ML methods can be derived as penalized estimators

$$\hat{\theta} = arg\,\min_{\theta} \Sigma_i L\{y_i - f(x_i, \theta)\} + \lambda R(\theta)$$

for loss function $L(.)$ and penalty function $R(.)$.

$R(\theta) = \|\|\theta\|\|_1$:	Lasso
$R(\theta) = \|\|\theta\|\|_2^2$:	Ridge regression
$R(\theta) = \alpha\|\|\theta\|\|_1 + (1-\alpha)\|\|\theta\|\|_2^2$:	Elastic net

- Penalty forces regularization: Well-behaved estimates, useful for prediction, even if $J > N$
- Regularization crucial for prediction performance

Predict monthly return of individual U.S. stocks with past returns

- Cross-section of $i = 1, \ldots, N$, with $Jx1$ characteristics vector (observable predictors) x_{it}.

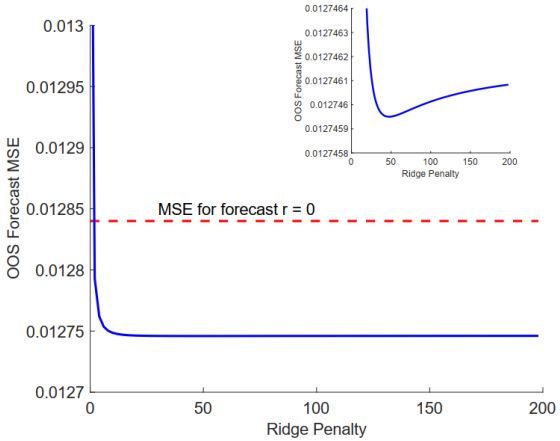
$$\mathbb{E}[r_{i,t+1}|x_{it}] = f(x_{it}, \theta)$$

- Observations $r_t = (r_{1t}, \ldots, r_{N,t})$ for $t = 1, \ldots, T$.
- x_{it} contains:
  - 120 lags of monthly returns, $r_{it}, r_{i,t-1}, r_{i,t-2}, \ldots r_{i,t-120}$
  - 120 lags of monthly squared returns $r^2_{it}, r^2_{i,t-1}, r^2_{i,t-2}, \ldots r^2_{i,t-120}$

  where all returns are cross-sectionally demeaned each month (i.e., cross-sectional focus) and x_{it} is standardized.
- Estimate during 1980-2000. Evaluate forecasts out-of-sample during 2001-2019.
- Ridge regression (where $\lambda = 0$ implements OLS)

$$\hat{\theta} = arg \min_{\theta} \Sigma_i (r_{i,t+1} - \theta' x_{i,t})^2 + \lambda \theta' \theta$$

OLS is really bad



Source: Nagel (2019)

DataScience Hub

Issues in applying ML to Asset Pricing

	Typical ML application	Asset pricing
Signal-to-noise	**Outcome observable** e.g. { hotdog, not hotdog }	**Very noisy observation** of outcome e.g. {high $\mathbb{E}[r]$, low $\mathbb{E}[r]$}
Big Data dimensions	N **and** J **big**	J **big**, N **not so much**
Sparsity	**Often sparse** e.g., some regions of image irrelevant	**Unclear**
Lucas critique	**Often not an issue** e.g. hotdogs don't change shape in response to image classification	**Investors learn from data and adapt**

Source: Nagel (2019)

Quest: describe cross-section with a few factors

- Multi-decade quest: Describe cross-section of N excess stock returns, $\mathbb{E}[r]$, with small number (K) of factor excess returns where factors are returns on portfolios constructed based on firm characteristics (size, momentum, . . .).
- Popular factor models are sparse in characteristics, e.g.: Fama-French 3-, 4-, 5-factor models
- But can a characteristics-sparse representation of the SDF be adequate?
  - Taking into account all anomalies that have been discovered
  - Plus potentially hundreds or thousands of additional stock characteristics, including interactions
  - High-dimensional problem!

THE JOURNAL OF FINANCE • VOL. LI, NO. 1 • MARCH 1996

Multifactor Explanations of Asset Pricing Anomalies

EUGENE F. FAMA and KENNETH R. FRENCH*

ABSTRACT

Previous work shows that average returns on common stocks are related to firm characteristics like size, earnings/price, cash flow/price, book-to-market equity, past sales growth, long-term past return, and short-term past return. Because these

. . . but now the list of factors is endless

Risk type		Description	Examples
Common (113)	**Financial** (46)	Proxy for aggregate financial market movement, including market portfolio returns, volatility, squared market returns, among others	Sharpe (1964): market returns; Kraus and Litzenberger (1976): squared market returns
	Macro (40)	Proxy for movement in macroeconomic fundamentals, including consumption, investment, inflation, among others	Breeden (1979): consumption growth; Cochrane (1991): investment returns
	Microstructure (11)	Proxy for aggregate movements in market microstructure or financial market frictions, including liquidity, transaction costs, among others	Pastor and Stambaugh (2003): market liquidity; Lo and Wang (2006): market trading volume
	Behavioral (3)	Proxy for aggregate movements in investor behavior, sentiment or behavior-driven systematic mispricing	Baker and Wurgler (2006): investor sentiment; Hirshleifer and Jiang (2010): market mispricing
	Accounting (8)	Proxy for aggregate movement in firm-level accounting variables, including payout yield, cash flow, among others	Fama and French (1992): size and book-to-market; Da and Warachka (2009): cash flow
	Other (5)	Proxy for aggregate movements that do not fall into the above categories, including momentum, investors' beliefs, among others	Carhart (1997): return momentum; Ozoguz (2009): investors' beliefs
Characteristics (202)	**Financial** (61)	Proxy for firm-level idiosyncratic financial risks, including volatility, extreme returns, among others	Ang et al. (2006): idiosyncratic volatility; Bali, Cakici, and Whitelaw (2011): extreme stock returns
	Microstructure (28)	Proxy for firm-level financial market frictions, including short sale restrictions, transaction costs, among others	Jarrow (1980): short sale restrictions; Mayshar (1981): transaction costs
	Behavioral (3)	Proxy for firm-level behavioral biases, including analyst dispersion, media coverage, among others	Diether, Malloy, and Scherbina (2002): analyst dispersion; Fang and Peress (2009): media coverage
	Accounting (87)	Proxy for firm-level accounting variables, including PE ratio, debt-to-equity ratio, among others	Basu (1977): PE ratio; Bhandari (1988): debt-to-equity ratio
	Other (24)	Proxy for firm-level variables that do not fall into the above categories, including political campaign contributions, ranking-related firm intangibles, among others	Cooper, Gulen, and Ovtchinnikov (2010): political campaign contributions; Edmans (2011): intangibles

Source: Factor List from Harvey et al. (2016)

Recent contributions

	Regularization	Assets	Nonlinearity
SDF models			
Kozak, Nagel, Santosh (2019)	elastic net	char. portfolios PC portfolios	interactions
Kozak (2019)	elastic net	char. portfolios PC portfolios	kernels
Giglio, Feng, and Xiu (2019)	Lasso	char. portfolios	-
DeMiguel et al. (2019)	Lasso	char. portfolios	-
Beta models			
Kelly, Pruitt, Su (2018)	PCA cutoff	indiv. stocks	-
Gu, Kelly and Xiu (2019)	Lasso	char. portfolios	autoencoder neural nets
Return prediction models			
Freyberger, Neuhierl, Weber (2018)	Group lasso	indiv. stocks	splines
Moritz and Zimmerman (2016)	Random forest	indiv. stocks	interactions
Gu, Kelly, Xiu (2018)	many	indiv. stocks	many

Source: Nagel (2019)

How to tackle this with ML?

- Penalize based on economic theory to reduce overfitting

$$\hat{b} = \arg\min_b (\hat{f} - \Sigma b)' \Sigma^{-} 1 (\hat{f} - \Sigma b) + \underbrace{\gamma_1 b' b}_{L2} + \underbrace{\gamma_2 \Sigma_{i=1}^{H} |b_i|}_{L1}$$

- L_1 en L_2 are regularization penalties and are based on economic theory
  - Sharpe ratio's can't be too big
  - Many of the covariates will be uninformative
- Summary of key results
  1. Shrinkage is extremely important
  2. Very little redundancy in original characteristics space: Characteristics-sparse SDF not achievable
  3. But PC-sparse SDF based on a few (high-variance) PCs prices well
- Result (2) could be partly a consequence of looking at a set of data-mined anomalies
- Could there be more characteristics-sparsity if we include some unexplored factors, or factors that are not known to be associated with return premia?

How does Big Data affect investors?

- See Martin and Nagel (2022, JFE) and Farboodi et al. (2022, RFS) for an excellent discussions
- Modern investors face a high-dimensional prediction problem: thousands of observable variables are potentially relevant for forecasting
- Framed as an ML problem, N assets have cash flows that are a (linear) function of J firm characteristics, but with uncertain coefficients
- Risk-neutral Bayesian investors impose shrinkage (Ridge regression) or sparsity (Lasso) when they estimate the J coefficients of the model for pricing assets
- When J is comparable in size to N, returns appear cross-sectionally predictable using firm characteristics to an econometrician who analyzes data from the economy ex post. A factor zoo emerges even without p-hacking and data-mining.
- Standard in-sample tests of market efficiency reject the no-predictability null with high probability, despite the fact that investors optimally use the information available to them in real time. In contrast, out-of-sample tests keep their economic meaning

Big Data and Investors *cont.*

- The economic content of the (semi-strong) market efficiency notion that prices "fully reflect" all public information is not clear in a high-dimensional setting
  - Abstracting from joint hypothesis problem Fama (1970, JoF): the econometrician studying asset prices does not know the model that determines risk premia required by risk-averse investors
- Does "fully reflect" mean that investors:
  1. know the parameters of the cash-flow prediction model $\rightarrow$ typical RE notion?
  2. employ Bayesian updating when they learn from data about the parameters of the cash-flow prediction model?
- The null hypothesis in a vast empirical literature in asset pricing is 1)
  - Literature on return predictability regressions, event studies, and asset pricing model estimation based on orthogonality conditions
- An apparent rejection of market efficiency == unsurprising consequence of investors not having precise knowledge of the parameters of a DGP that involves thousands of predictor variables

Will your new approach be implemented?

- Is there potential "Alpha content"?
  - Does the new data or method give rise to sufficient risk-adjusted return to merit implementation of a stand-alone strategy or as a component of a portfolio strategy (cf Kolanovic and Krishnamachari (2017))
- Markets already digest a lot of information .... so the room for improvement is small

"The flat maximum effect states that for most problems there is not a single best model that is substantially better than all others." *(Finlay (2014), page 105)*

The markets' view: J.P. Morgan

- Kaggle suggests that structured data is best analyzed by tools like XGBoost and Random Forests
- Use of Deep Learning is limited to analysis of images or text
  - Deep Learning tools still require a substantial amount of data to train. Training on small sample sizes (e.g. generative-adversarial models) is still at an early stage
  - Large sample data required implies that first applications of Deep Learning will be in intraday or high-frequency trading before we see its application in lower frequencies (See Algorithmic Trading course!!!)
- Deep Learning finds immediate use for portfolio managers in an indirect manner. Parking lot images are analyzed using Deep Learning architectures (like CNN) to count cars. Text in social media is analyzed using Deep Learning architectures (like LSTM) to detect sentiment
- Such traffic and sentiment signals can be integrated directly into quantitative strategies (See Kolanovic and Krishnamachari (2017))
- Calculation of signals often outsourced to specialized firms

Summary

In this lecture we covered:

1. We looked at methods to reduce the complexity to models
   - Dimension reduction through PCA
   - Feature selection with RIDGE and LASSO
2. We also discussed ways of assessing how well a model performs
   - holdout
   - K-fold cross validation
3. An application to asset pricing

References I

- Anscombe, F. (1973). Graphs in Statistical Analysis. American Statistician, 27, 17–21.
- Bianchi, D., Büchner, M., Hoogteijling, T., & Tamoni, A. (2021). Corrigendum: Bond Risk Premiums with Machine Learning. The Review of Financial Studies, 34, 1090–1103.
- Bianchi, D., Büchner, M., & Tamoni, A. (2021). Bond Risk Premiums with Machine Learning [Publisher: Oxford Academic]. The Review of Financial Studies, 34(2), 1046–1089.
- Chen, A. Y. (2021). The Limits of p-Hacking: Some Thought Experiments [Publisher: John Wiley & Sons, Ltd]. The Journal of Finance, 76(5), 2447–2480.
- Easley, D., López De Prado, M., O'hara, M., & Zhang, Z. (2021). Microstructure in the Machine Age. The Review of Financial Studies, 34, 3316–3363.
- Erel, I., Stern, L. H., Tan, C., & Weisbach, M. S. (2021). Selecting Directors Using Machine Learning. Review of Financial Studies, 34, 3226–3264.

References II

- Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks [Publisher: Nature Publishing Group]. Nature, 542(7639), 115–118.
- Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. The Journal of Finance, 25(2), 383–417.
- Farboodi, M., Matray, A., Veldkamp, L., & Venkateswaran, V. (2022). Where Has All the Data Gone? Review of Financial Studies, 35(7), 3101–3138.
- Finlay, S. (2014). Predictive Analytics, Data Mining and Big Data. Palgrave MacMillan.
- Fuster, A., Goldsmith-Pinkham, P., Ramadorai, T., & Walther, A. (2022). Predictably Unequal? The Effects of Machine Learning on Credit Markets. Journal of Finance, 77(1), 5–47.
- Goldstein, I., Spatt, C. S., & Ye, M. (2021). Big Data in Finance. Review of Financial Studies, 34, 3213–3225.

References III

- Harvey, C. R., Liu, Y., & Zhu, H. (2016). . . . and the Cross-Section of Expected Returns [Publisher: Narnia]. Review of Financial Studies, 29(1), 5–68.
- Hastie, T., Tibshirani, R., & Friedman, J. (2017). The Elements of Statistical Learning: Data Mining, Inference, and Prediction [arXiv: 1011.1669v3 ISSN: 03436993].
- Kolanovic, M., & Krishnamachari, R. T. (2017). Big Data and AI Strategies - Machine Learning and Alternative Data Approach to Investing Quantitative and Derivatives Strategy. J.P. Morgan report.
- Kozak, S., Nagel, S., & Santosh, S. (2019). Shrinking the Cross Section. Journal of Financial Economics.
- Leippold, M., Wang, Q., & Zhou, W. (2022). Machine learning in the Chinese stock market. Journal of Financial Economics.
- Li, K., Mai, F., Shen, R., & Yan, X. (2021). Measuring Corporate Culture Using Machine Learning. The Review of Financial Studies, 34, 3265–3315.

References IV

- Martin, I., & Nagel, S. (2022). Market Efficiency in the Age of Big Data. Journal of Financial Economics, 145(1), 154–177. http://www.nber.org/papers/w26586
- Nagel, S. (2019). Asset Pricing and Machine Learning - Lecture 1. Princeton Lectures in Finance.
- Obaid, K., & Pukthuanthong, K. (2022). A picture is worth a thousand words: Measuring investor sentiment by combining machine learning and photos from news. Journal of Financial Economics.
- Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Langlotz, C., Shpanskaya, K., Lungren, M. P., & Ng, A. Y. (2017). CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning [arXiv:1711.05225 [cs, stat]].
- Yang, L., & Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. Neurocomputing, 415, 295–316.