

ML for SS

Dr. Richard M. Crowley

Purpose of this course

This course explores a wide swath of machine learning tools that are both 1) practical for social science research and 2) readily available to implement. After this course, students should be well prepared to implement certain machine learning approaches into empirical research projects, and well prepared to explore other methods that are not directly covered.

Course structure

Each week will cover a different methodology, either a machine learning technique or a more fundamental technique upon which machine learning methods can be applied. Lessons will consist of two parts: a review of two to three papers that utilize the week's methodology (usually applied papers, occasionally review papers or more technical papers), an overview of how the methodology works, and a worked out example of implementing the methodology. From Session 2 onward, paper discussions will largely be student led, while methodology overview and application will be handled by the professor.

Expectations:

1. Keep an open mind. This course covers a wide swath of disciplines, not just because you will be coming from a variety of programs, but also because different applicable methodologies are more commonly used (or better used) in different disciplines. As such, required readings are pulled from Accounting, Archaeology, Computer Science, Economics, Health Science, Finance, Marketing, Operations Management, Political Science, Psychology, and Statistics. For each paper, even if you are not familiar with some of the constructs the papers use, the importance and usefulness of the methodology used should be transparent.
2. You should come prepared to discuss the required readings for each week. Optional readings are supplied and are intended to be read by anyone with a keen interest in applying that week's methodology.
3. You should enter the class with at least a remedial understanding of Python or R coding¹. There is no formal prerequisite in this regard, but if you are not sure if you are prepared

¹Homeworks and the final project can be completed in other languages, but any in class examples will be done in Python 3 and/or R as fits the algorithm and class composition. Much of what we will cover can also be done reasonably easily in either language. C or Java will work fine as well, should you be so inclined, but those will require more coding effort. If you wish to use any other language on homeworks, please clear it with me first.

well enough, I would recommend you check out the following online classes on Datacamp for Python: 1) [Intro to Python for Data Science](#) and [Intermediate Python](#). For those interested in R, you can check out Datacamp's 1) [Introduction to R](#) and [Introduction to Tidyverse](#).

Pre-requisites

Students are required to have a working knowledge of either R or python. This can be satisfied through any of the following ways:

1. Having taken a formal course on either programming language.
2. Having taken an online course (such as via Datacamp) on either programming language.
3. Having self-studied R or python language enough to be comfortable coding in it, e.g., having self-studied the language in order to implement a research project.

For students who have yet to fulfill any of the three, I would recommend you take the Datacamp courses mentioned in Expectations point 3 above *before* the class starts.

Presentations

To facilitate discussion in each class session, two to three papers will each be presented by a pair of students. Every student taking the course for a grade is expected to present at least twice, though overall presentation counts will depend on course enrollment. As a presenter, you should not simply present what the paper does, but also consider the following:

1. Why does the paper use the methodology it does?
 - Is it because the methodology is optimal? Good enough? Publishable?
2. What could the paper do better?

You are encouraged to also prepare a few discussion questions for the class to discuss or debate.

Pre-class summaries

These are **optional**. However, they may be made mandatory at the professor's discretion if class discussion is not sufficiently vibrant. Should they be made mandatory, the policy will be as follows:

Pre-class summaries are designed to encourage deeper thinking about each paper we read for the course. This also encourages you to have questions prepared when you come to class, so as to lead to a richer paper discussion. For the summary you will need to think about both the theoretical relationship the paper is trying to capture as well as the way in which this relationship is operationalized. You will also be asked to push beyond the paper, by either asking questions about what the paper is doing, or by suggesting extensions for the paper or its methodology.

A template is available on eLearn for this. Summaries will be graded on a three-tiered system akin to pass-fail: A check represents a sufficient response. For a response that is very well thought out, a check plus will be given. For an insufficient response, a check minus will be given.

Assignments

Throughout the course, there will be three assignments to illustrate how different algorithms work. These will require you to apply the methodology to a provided data set using your preferred programming language (python or R is recommended). Note that while statistical software such as SAS, Stata, and SPSS may be able to do some of the simpler methodologies, they are often much more difficult to use for the methodologies we will be analyzing. Additionally, some assignments will ask you to explain your code. Assignment can be completed in teams of 2.

Assignments can be done in any of the following ways:

1. For an **all python** workflow, consider using a [jupyter notebook](#). This will allow you to put all code, explanations, and output in the same file.
2. For an **all R** workflow, consider using [Quarto](#) from Rstudio. This will allow you to put all code, explanations, and output in the same file.
3. For **hybrid** workflows, you can use [reticulate](#) to run python code within Rmarkdown, [rpy2](#) to interact with R from a python kernel in jupyter, or just submit separate Rmarkdown and jupyter files. You can also use [Quarto](#) from either Jupyter or Rstudio.
4. Make separate files for code and for results and explanations, though I would encourage you to explore the above methods first.

Final project: Proposal

The proposal will require you to think about how machine learning techniques can help us to approach new questions or better explore questions that have been previously examined. As part of the proposal, you will be required to use tools from at least *two* of the four main topics covered throughout the course (ML Regression, Text analytics, ML and econometrics, and Neural networks). The proposal will be due a few weeks after the final session of the class. Precise details for the proposal are given in the Proposal Instructions document.

Grading

The following weights apply for course grading:

	Item	Weight
	Class presentations	20%
	Participation	20%
	Assignments (equally weighted)	30%
	Final project	30%

Topics (Subject to update in August 2025)

Note

All materials are in the order that I would recommend reading them in. Optional materials are indicated as such.

Subjects and papers covered are subject to change, based on student feedback and response at the professor's discretion.

New for 2024: 14 new papers; 2 new subject areas!

ML regression

Week 1: General workflow; ML regression

Data · “Big” data · Training/Validation/Testing · LASSO/Elastic net

Required reading

1. Mullainathan, Sendhil, and Jann Spiess. “Machine learning: an applied econometric approach.” *Journal of Economic Perspectives* 31, no. 2 (2017): 87-106.
 - A nice overview of what we will be covering.
2. Chahuneau, Victor, Kevin Gimpel, Bryan R. Routledge, Lily Scherlis, and Noah A. Smith. “Word salad: Relating food prices and descriptions.” In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1357-1367. 2012.
 - A simple use of LASSO for understanding menu pricing at restaurants.

Optional reading

- Tibshirani, Robert. “Regression shrinkage and selection via the lasso.” *Journal of the Royal Statistical Society: Series B (Methodological)* 58, no. 1 (1996): 267-288.
 - Technical details for LASSO.
- Meinshausen, Nicolai. “Relaxed lasso.” *Computational Statistics & Data Analysis* 52, no. 1 (2007): 374-393.
 - A bit of an improvement on LASSO called “Relaxed LASSO” ([How to fit with R and glmnet](#)).
 - Some evidence that it is fairly optimal for both low and high signal to noise ratio problems: Hastie, Trevor, Robert Tibshirani, and Ryan Tibshirani. “Best Subset, Forward Stepwise or Lasso? Analysis and Recommendations Based on Extensive Comparisons.” *Statistical Science* 35, no. 4 (2020): 579-592.
- Belloni, Alexandre, and Victor Chernozhukov. “Least squares after model selection in high-dimensional sparse models.” (2013): 521-547.

- A method for generating statistical inferences and p-values from LASSO.
- Jaakonmäki, Roope, Oliver Müller, and Jan Vom Brocke. “The impact of content, context, and creator on user engagement in social media marketing.” In Proceedings of the 50th Hawaii international conference on system sciences. 2017.
 - A simple application of LASSO for marketing.

R coding resources

- R has robust econometric tools built in, and many other available. For HDFE, [fixest](#) is extremely efficient and flexible.
- For LASSO and elastic net [GLMnet](#) is probably the best and is fairly easy to use (including a very efficient cross-validated method); [caret](#) also works.
- For out-of-sample time series prediction, Facebook’s [Prophet](#) library is available for R. It runs on Stan, which is perhaps the most well-known Bayesian inference library.

Python coding resources

- For econometrics in python, [Statsmodels](#), [linearmodels](#), and [scikit-learn](#) are useful.
- For LASSO, use [sklearn.linear_model.Lasso](#) or [sklearn.linear_model.LassoCV](#) (for automated cross-validation).
- For elastic net, use [sklearn.linear_model.ElasticNet](#) or [sklearn.linear_model.ElasticNetCV](#) (for automated cross-validation).
- For out-of-sample time series prediction, Facebook’s [Prophet](#) library is also available for Python.

Week 2: Classification

SVM · Tree-based classifications (e.g., XGBoost)

Required reading

1. Purda, Lynnette, and David Skillicorn. “Accounting variables, deception, and a bag of words: Assessing the tools of fraud detection.” *Contemporary Accounting Research* 32, no. 3 (2015): 1193-1223.
 - Classification of misreporting with SVM.
2. Noh, Byungjoo, Changhong Youm, Eunkyong Goh, Myeounggon Lee, Hwayoung Park, Hyojeong Jeon, and Oh Yoen Kim. “XGBoost based machine learning approach to predict the risk of fall in older adults using gait outcomes.” *Scientific reports* 11, no. 1 (2021): 12183.
 - Using XGBoost and experimental data to build a fall prediction algorithm for older adults

Optional reading

- Bao, Yang, Bin Ke, Bin Li, Y. Julia Yu, and Jie Zhang. “Detecting accounting fraud in publicly traded US firms using a machine learning approach.” *Journal of Accounting Research* 58, no. 1 (2020): 199-235.
 - Applies RUSBoost to accounting data to predict accounting fraud.
- Chen, Tianqi, and Carlos Guestrin. “Xgboost: A scalable tree boosting system.” In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785-794. 2016.
 - The technical details behind XGBoost.
- [Lecture slides](#) from my Hong Kong Shue Yan RGC IIDS talk
 - Covers an XGBoost extension of Brown, Nerissa C., Richard M. Crowley, and W. Brooke Elliott. “What are you saying? Using topic to detect financial misreporting.” *Journal of Accounting Research* 58, no. 1 (2020): 237-291.
 - * This paper is closely related to Purda and Skillicorn (2015)
 - All the R code to replicate the analysis is also available at the link!

R coding resources

- For SVM you can use the venerable [e1071](#) library or use [kernlab](#).
- For XGBoost, there is the [xgboost](#) library.
- For other options, consider [LightGBM](#) and [randomForest](#).

Python coding resources

- For SVM and simpler tree methods (random forest, GBM), [scikit-learn](#) works well.
- The original [xgboost](#) package is available for Python.
 - As compared to the R package, it has minimal support for categorical information, but supports GPU workflows and larger-than-memory calculation.
- Other options for similar methods include [LightGBM](#).

Week 3: Clustering and ensembling

KNN · T-SNE · UMAP | Voting · Bagging · Boosting

Required reading

1. Easton, Peter D., Martin M. Kapons, Steven J. Monahan, Harm H. Schütt, and Eric H. Weisbrod. “Forecasting earnings using K-nearest neighbors.” *The Accounting Review* 99, no. 3 (2024): 115-140.
 - Analyst forecasts using KNN.
2. Noichl, Maximilian. “How localized are computational templates? A machine learning approach.” *Synthese* 201, no. 3 (2023): 107.

Optional reading

- Hie, Brian, Ellen D. Zhong, Bonnie Berger, and Bryan Bryson. “Learning the language of viral evolution and escape.” *Science* 371, no. 6526 (2021): 284-288.
 - Contains some great use of UMAP to explain high-dimensional data.
- Lee, Thomas Y., and Eric T. Bradlow. “Automated marketing research using online customer reviews.” *Journal of Marketing Research* 48, no. 5 (2011): 881-894.
 - Uses kmeans clustering to group online reviews.
- Qiu, Yue, Tian Xie, and Jun Yu. “Forecast combinations in machine learning.” (2020)
 - A straightforward use of Ensemble methods in economics.
- Wang, Bingling, Min-Bin Lin, and Wolfgang Karl Härdle. “NFTs and VizTech.” Available at SSRN 4490759 (2023).
 - Uses t-SNE and UMAP to visualize cryptopunk clustering

R coding resources

- For clustering, [caret](#) works well.
- For T-SNE, [Rtsne](#) works well.
- For UMAP, [umap](#) can be used either purely in R or through calling the python [umap-learn](#) package
- For Ensembling, you can use [SuperLearner](#), [EnsembleML](#), or roll your own.

Python coding resources

- For basic clustering, T-SNE, and most any ensembling, [scikit-learn](#) works well.
- For UMAP, the [umap-learn](#) package is recommended. It can be a bit clunky, but the UMAP implementation is much more efficient than T-SNE.

ML and econometrics

Week 4: Causal ML and non-causal ML

Double/debiased/neyman ML | Policy Problems

Machine learning techniques can be used to help augment our traditional approaches used in making causal inferences.

Required reading

1. Chernozhukov, Victor, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, and Whitney Newey. “Double/debiased/neyman machine learning of treatment effects.” *American Economic Review* 107, no. 5 (2017): 261-65.
 - **Read the Web Appendix as well!**
 - The web appendix contains the applied part of the paper, and is a much easier read than the paper itself.
 - An influential new method for using machine learning for causality.
2. Crowley, Richard M., Yun Lou, Samuel T. Tan, and Liandong Zhang. “Misinformation regulations: Early evidence on corporate social media strategy.” *Review of Accounting Studies* (2025): 1.
 - [Early online link](#)
 - Implements two DoubleML models to help illustrate potential causality of its findings.
3. Kleinberg, Jon, Jens Ludwig, Sendhil Mullainathan, and Ziad Obermeyer. “Prediction policy problems.” *American Economic Review* 105, no. 5 (2015): 491-95.
 - A short overview of when prediction is more important than causation in policy problems, along with an explanation and example of how to solve such problems.
4. Chalfin, Aaron, Oren Danieli, Andrew Hillis, Zubin Jelveh, Michael Luca, Jens Ludwig, and Sendhil Mullainathan. “Productivity and selection of human capital with machine learning.” *American Economic Review* 106, no. 5 (2016): 124-27.
 - Short paper illustrating a simple use case of policy prediction.

Optional reading

- Chernozhukov, Victor, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. “Double/debiased machine learning for treatment and structural parameters.” *The Econometrics Journal* 21, no. 1 (2018): C1-C68.
 - The technical details behind the Double/debiased/neyman ML paper.
- Chang, Neng-Chieh. “Double/debiased machine learning for difference-in-differences models.” *The Econometrics Journal* 23, no. 2 (2020): 177-191.
 - A DID tweak of the DoubleML model

- Knaus, Michael C. “A double machine learning approach to estimate the effects of musical practice on student’s skills.” *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 184, no. 1 (2021): 282-300.
 - An applied paper using DoubleML.
- Athey, Susan, and Guido W. Imbens. “The state of applied econometrics: Causality and policy evaluation.” *Journal of Economic Perspectives* 31, no. 2 (2017): 3-32.
 - A review paper with a focus on policy evaluation and a bit on causal methods as well.
- Athey, Susan. “The impact of machine learning on economics.” In: *The economics of artificial intelligence: An agenda*, pp. 507-547. University of Chicago Press, 2018.
 - A forward looking review of ML in Economics.
- Deryugina, Tatyana, Garth Heutel, Nolan H. Miller, David Molitor, and Julian Reif. “The mortality and medical costs of air pollution: Evidence from changes in wind direction.” *American Economic Review* 109, no. 12 (2019): 4178-4219.
 - Applies XGBoost for classifying mortality as it relates to air pollution.
 - Note: What this paper calls CDDF is related to the DoubleML method.
 - The paper is very careful in its execution – good to study to see how to push causality methods in the empirical literature forward.

R coding resources

- Both [MCKnaus/dmlmt](#) and [DoubleML/doubleml-for-r](#) are worth looking into. The latter is fairly easy to use, and is based on the Chernozhukov et al. papers.
- The original R code for the Chernozhukov et al. (2017) AER paper is available from AER
 - The code is quite clean and extensible

Python coding resources

- [DoubleML](#) is easy to use to do Chernozhukov et al. (2017) style tests, while [microsoft/EconML](#) provides a number of other related models.

Week 5: Bias and fairness

SHAP · Custom estimators

Required reading

1. Wich, Maximilian, Jan Bauer, and Georg Groh. “Impact of politically biased data on hate speech classification.” In Proceedings of the Fourth Workshop on Online Abuse and Harms, pp. 54-64. 2020.
 - A paper using SHAP to understand an impact of political bias.
2. Gentzkow, Matthew, Jesse M. Shapiro, and Matt Taddy. “Measuring group differences in high-dimensional choices: method and application to congressional speech.” *Econometrica* 87, no. 4 (2019): 1307-1340.
 - Another method focused on improving causality using machine learning.

Optional reading

- Lundberg, Scott, and Su-In Lee. “A unified approach to interpreting model predictions.” In Proceedings of the 31st Conference on Neural Information Processing Systems. (2017).
 - The technical details behind SHAP
 - This paper will be covered in class by the professor up front
- Rambachan, Ashesh, Jon Kleinberg, Jens Ludwig, and Sendhil Mullainathan. “An economic perspective on algorithmic fairness.” In *AEA Papers and Proceedings*, vol. 110, pp. 91-95. 2020.
 - A high level overview of bias from an algorithmic and economics perspective.
- Shapley, Lloyd S. “A value for n-person games.” *Contributions to the Theory of Games* 2, no. 28 (1953): 307-317.
 - The game theory foundation of the SHAP algorithm.

R coding resources

- [ModelOriented/shapper](#) provides an implementation of SHAP – good for calculating SHAP values, but mostly requires you to roll your own visualizations
- [bgreenwell/fastshap](#) is focused on efficient calculations of SHAP values
- [ModelOriented/DALEX](#)
- [liuyanguu/SHAPforxgboost](#) is useful for explicitly applying SHAP to XGBoost

Python coding resources

- Python: [slundberg/shap](#) Is good for both calculating SHAP values and visualizing the output. It is flexible enough to work across most model types from XGBoost to neural networks.

Text analytics

Week 6: Basic natural language processing (NLP) and sentiment

Sentiment dictionaries · Supervised classifiers

Required reading

1. Loughran, T. and McDonald, B., 2011. When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. *The Journal of Finance*, 66(1), pp.35-65.
 - Perhaps the most influential sentiment paper in Finance and Accounting for the past decade. Where the Loughran McDonald dictionaries come from.
2. Hassan, Tarek A., Stephan Hollander, Laurence Van Lent, and Ahmed Tahoun. “Firm-level political risk: Measurement and effects.” *The Quarterly Journal of Economics* 134, no. 4 (2019): 2135-2202.
 - Another approach to classifying text, but one that is less manual.

Optional reading

- Antweiler, Werner, and Murray Z. Frank. “Is all that talk just noise? The information content of internet stock message boards.” *The Journal of Finance* 59, no. 3 (2004): 1259-1294.
 - A seminal paper in textual analysis in Finance that uses Naive Bayes to classify sentiment.
- Loughran, Tim, and Bill McDonald. “Measuring firm complexity.” *Journal of Financial and Quantitative Analysis* (2023): 1-28.
 - A more recent dictionary-oriented study by Loughran and McDonald.
- Gentzkow, Matthew, Bryan Kelly, and Matt Taddy. “Text as data.” *Journal of Economic Literature* 57, no. 3 (2019): 535-74.
 - A summary of textual data for economic research.
- Hassan, Tarek A., Stephan Hollander, Laurence Van Lent, Markus Schwedeler, and Ahmed Tahoun. “Firm-level exposure to epidemic diseases: Covid-19, SARS, and H1N1.” *The Review of Financial Studies* 36, no. 12 (2023): 4919-4964.
 - A very straight-forward application of textual data that is quite timely.
- Das, Sanjiv R., and Mike Y. Chen. “Yahoo! for Amazon: Sentiment extraction from small talk on the web.” *Management science* 53, no. 9 (2007): 1375-1388.
 - Another seminal paper along the lines of the Antweiler and Frank paper.
- Loughran, Tim, and Bill McDonald. “Textual Analysis in Finance.” *Annual Review of Financial Economics* 12 (2020): 357-375.
 - A nice and current review paper that takes a somewhat critical look at some of the textual analysis work that has been done in finance and accounting.

R coding resources

- For general text processing, [tidytext](#) is very helpful.
- For regular expressions, consider using [stringr](#) instead of Base R.
- For simple calculations (sentiment, readability, etc.), [quanteda](#) can handle these easily.
- For supervised classifiers, [caret](#) and [e1071](#) are good choices

Python coding resources

- A lot of traditional data sources and analytics for NLP are included in [NLTK](#).
- A more modern approach (built on machine learning) is [spaCy](#). Often times spaCy's approach is more accurate, but it doesn't have as many functions as NLTK.
- For supervised classifiers, [scikit-learn](#) has many choices.

Week 7: Linguistics

Grammatical structure · POS Tagging · Word embeddings

Required reading

1. Harris, Zellig S. “Distributional structure.” *Word* 10, no. 2-3 (1954): 146-162.
 - The insights of this paper underlie most modern computational approaches to working with text (even LLMs)
2. Garimella, Aparna, Carmen Banea, Dirk Hovy, and Rada Mihalcea. “Women’s syntactic resilience and men’s grammatical luck: Gender-bias in part-of-speech tagging and dependency parsing.” In *Association for Computational Linguistics*. 2019.
 - Examines differences in grammar across men and women using a hand-annotated corpus.
3. Garg, Nikhil, Londa Schiebinger, Dan Jurafsky, and James Zou. “Word embeddings quantify 100 years of gender and ethnic stereotypes.” *Proceedings of the National Academy of Sciences* 115, no. 16 (2018): E3635-E3644.
 - Illustrates a way to analyze text to examine historical demographic and sociological trends.

Optional reading

- Jurafsky, Dan, Victor Chahuneau, Bryan R. Routledge, and Noah A. Smith. “Narrative framing of consumer sentiment in online restaurant reviews.” *First Monday* (2014).
 - Examines “narrative framing” in reviews of restaurants online, using a fairly simple approach.
- Hope, Ole-Kristian, Danqi Hu, and Hai Lu. “The benefits of specific risk-factor disclosures.” *Review of Accounting Studies* 21, no. 4 (2016): 1005-1045.
 - Examines accounting disclosure specificity using NER.
- Hanley, Kathleen Weiss, and Gerard Hoberg. “Dynamic interpretation of emerging risks in the financial sector.” *The Review of Financial Studies* 32, no. 12 (2019): 4543-4603.
 - Uses word embeddings to automate dictionary construction
- Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. “Efficient estimation of word representations in vector space.” *arXiv preprint arXiv:1301.3781* (2013).
 - Technical details for word2vec.

R coding resources

- For grammar parsing, consider [spacyr](#) or [coreNLP](#).
- For word embeddings, you can use [word2vec](#)

Python coding resources

- [NLTK](#) has a number of parsers built in, mostly statistical in nature
- [spaCy](#) has machine learning pipelines for part of speech tagging, dependency parsing, and named entity recognition.
- [Stanford NLP](#) also offers a text parser; the core functions are available in python using [Stanza](#). Some useful and more specialized functions are only available via Java, however.
- For word2vec, [gensim](#) is both easy to use and fast to run.
 - Note: It has some incompatibilities with Python 3.9 (as of July 2021).
- For another model similar to word2vec, consider Meta's [fastText](#), which has many different languages available.

Week 8: Topic modeling

LDA · STM · Embedded Topic Modeling

Required reading

1. Brown, Nerissa C., Richard M. Crowley, and W. Brooke Elliott. “What are you saying? Using topic to detect financial misreporting.” *Journal of Accounting Research* 58, no. 1 (2020): 237-291.
 - Applying LDA to annual report text in order to predict corporate misreporting. Includes extensive validation of LDA for annual reports.
2. Roberts, M.E., Stewart, B.M., Tingley, D., Lucas, C., Leder-Luis, J., Gadarian, S.K., Albertson, B. and Rand, D.G., 2014. Structural topic models for open-ended survey responses. *American Journal of Political Science*, 58(4), pp.1064-1082.
 - A more extensible LDA model that allows one to impose a regression-like structure on them.
3. Dieng, Adji B., Francisco JR Ruiz, and David M. Blei. “Topic modeling in embedding spaces.” *Transactions of the Association for Computational Linguistics* 8 (2020): 439-453.
 - This model combines word2vec and LDA into a single model. More interestingly, it highlights statistical limitations of many traditional ML models and demonstrates what can be done to bring better statistical foundations into machine learning methods.

Optional reading

- Blei, David M., Andrew Y. Ng, and Michael I. Jordan. “Latent dirichlet allocation.” *the Journal of Machine Learning research* 3 (2003): 993-1022.
 - Technical details for LDA. This is effectively a statistics paper, with a bit of machine learning just to solve the statistics problem.
- Huang, Allen H., Reuven Lehavy, Amy Y. Zang, and Rong Zheng. “Analyst information discovery and interpretation roles: A topic modeling approach.” *Management Science* 64, no. 6 (2018): 2833-2855.
 - A nice conceptual application of LDA.

R coding resources

- [stm](#) (Structural Topic Models) is a nice twist on LDA.
- [lda](#), [topicmodels](#), and [RMallet](#) work for basic LDA implementations.
- ETM can be done using [topicmodels.etm](#).

Python coding resources

- For LDA, [gensim](#) is both easy to use and fast to run.
 - Note: It has some incompatibilities with Python 3.9 (as of July 2021).
- Note that there is no STM implementation in python
- For ETM, you can consider [octis](#), [embedded-topic-model](#), or the [original repository](#) from the paper.

Neural networks

Week 9: Transformer Neural Networks

Transformers

Required reading

1. Crowley, Richard M., and Wong, Franco. “Understanding Sentiment through Context.” Working paper (2025).
 - I will provide this paper during the term break, as it is under active revision.
 - Explores leveraging topic modeling to better make sense of sentiment in a financial context.
2. Fritsch, Felix, Qi Zhang, and Xiang Zheng. “Responding to Climate Change Crises: Firms’ Trade-Offs.” *Journal of Accounting Research* (2025).
3. Assael, Yannis, Thea Sommerschild, Brendan Shillingford, Mahyar Bordbar, John Pavlopoulos, Marita Chatzipanagiotou, Ion Androutsopoulos, Jonathan Prag, and Nando de Freitas. “Restoring and attributing ancient texts using deep neural networks.” *Nature* 603, no. 7900 (2022): 280-283.
 - Using a transformer model to transcribe and date ancient texts.

Optional reading

- Grootendorst, Maarten. “BERTopic: Neural topic modeling with a class-based TF-IDF procedure.” arXiv preprint arXiv:2203.05794 (2022).
 - Technical details behind BERTopic. Mostly an applied CS approach.
- Huang, Allen H., Hui Wang, and Yi Yang. “FinBERT: A large language model for extracting information from financial text.” *Contemporary Accounting Research* 40, no. 2 (2023): 806-841.
 - Applies BERT (a transformer-based embedding model) with a bit of transfer learning for sentiment, ESG classification, and forward looking statement classification in finance.
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. “Attention is all you need.” *Advances in neural information processing systems* 30 (2017).
 - Technical details behind the transformer architecture

R coding resources

- Just like with Python, we can set up neural networks using [Keras](#) and run them with [TensorFlow](#) as the backend.
- [OpenIE](#) is available as a standalone Java program.

Python coding resources

- [Keras](#) is a good interface to use to set up neural networks, and [TensorFlow](#) is a good backend
- [pytorch](#) is another popular backend for neural networks in Python
- [Hugging Face](#) is good for using pretrained text models
 - [FinBERT is available here](#).
- [BERTopic](#) is a topic modeling approach leveraging word order in its assignment of topics.
- [tensorflow_hub](#) is good for implementing pretrained models for a variety of problems
- [Theano](#), while deprecated, is still common to see in older pretrained models. It is being superseded by [PYMC3](#).
- Code for [Ithaca on Github](#)

Week 10: Understanding Large Language Models (LLMs)

LLMs

Required reading

1. Sclar, M., Choi, Y., Tsvetkov, Y., & Suhr, A. Quantifying Language Models' Sensitivity to Spurious Features in Prompt Design or: How I learned to start worrying about prompt formatting. In The Twelfth International Conference on Learning Representations (2024).
 - Examines the stability of LLM output with respect to variation in prompts.
 - Think about what this means for social science research that makes use of LLMs.
2. Zhao, Yuan. "The Role of Fact-Checkers in Capital Markets." Working paper (2025).
 - Utilizes GPT models for significant amounts of data analysis and categorization. This will be presented by the author.
3. Levy, Bradford. "Caution ahead: Numerical reasoning and look-ahead bias in AI models." Available at SSRN 5082861 (2025).
 - A well argued paper cautioning against certain pitfalls of using LLMs.

Optional reading

- West, Peter, Ximing Lu, Nouha Dziri, Faeze Brahman, Linjie Li, Jena D. Hwang, Liwei Jiang et al. "THE GENERATIVE AI PARADOX: "What It Can Create, It May Not Understand"" In The Twelfth International Conference on Learning Representations. 2024.
 - Examines a counterintuitive issue of LLMs: they don't understand their own output well.

R coding resources

N/A

Python coding resources

- You can use GPT via the `openai` package
- You can use certain other LLMs such as LLAMA through the `huggingface` package

Week 11: Using Large Language Models (LLMs)

LLMs

Required reading

1. de Kok, Ties. “ChatGPT for Textual Analysis? How to use Generative LLMs in Accounting Research.” *How to use Generative LLMs in Accounting Research* (March 1, 2024) (2024).
 - A guide to using GPT models in accounting research
2. Nozza, Debora, Federico Bianchi, and Dirk Hovy. “HONEST: Measuring hurtful sentence completion in language models.” In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2021.
 - Understanding variation of hurtful language produced by transformer and LLM models across a variety of languages.
3. Plaza-del-Arco, Flor Miriam, Amanda Cercas Curry, Alba Curry, Gavin Abercrombie, and Dirk Hovy. “Angry Men, Sad Women: Large Language Models Reflect Gendered Stereotypes in Emotion Attribution.” *arXiv preprint arXiv:2403.03121* (2024).
 - Understanding gender bias inherent in LLMs.

Optional reading

- Ziems, Caleb, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. “Can large language models transform computational social science?.” *Computational Linguistics* 50, no. 1 (2024): 237-291.
 - A literature review of computational social science, and the extent to which LLMs may be able to contribute.
- Bernard, Darren, Elizabeth Blankespoor, Ties de Kok, and Sara Toynbee. “A Modular Measure of Business Complexity.” Available at SSRN 4480309 (2023).
 - An accounting-heavy GPT-based model for classifying business complexity.
- Rooein, Donya, Paul Rottger, Anastassia Shaitarova, and Dirk Hovy. “Beyond Flesch-Kincaid: Prompt-based Metrics Improve Difficulty Classification of Educational Texts.” *arXiv preprint arXiv:2405.09482* (2024).
 - An interesting use of LLM prompting as the level of analysis, which also includes an experimental component.

R coding resources

N/A

Python coding resources

- Ties de Kok's [companion website](#) on using Chat GPT in accounting research.
- You can use GPT via the `openai` package
- [Code](#) for the first paper.
- [Code and data](#) for the second paper.
- [Code and data](#) for the third paper.
- You can use GPT via the `openai` package
- You can use certain other LLMs such as LLAMA through the `huggingface` package

Week 12: Images: Classification and algorithms

Convolutional neural networks

Required reading

1. Zhang, Shunyuan, Dokyun Lee, Param Vir Singh, and Kannan Srinivasan. “What makes a good image? Airbnb demand analytics leveraging interpretable image features.” *Management Science* 68, no. 8 (2022): 5644-5666.
 - Examines how images in listings impact AirBNB properties
2. Aubry, Mathieu, Roman Kraeussl, Gustavo Manso, and Christophe Spaenjers. “Biased auctioneers.” *The Journal of Finance* 78, no. 2 (2023): 795-833.
 - Examines estimation errors in auction listings for art.
3. Khachiyani, Arman, Anthony Thomas, Huye Zhou, Gordon Hanson, Alex Cloninger, Tajana Rosing, and Amit K. Khandelwal. “Using neural networks to predict microspatial economic growth.” *American Economic Review: Insights* 4, no. 4 (2022): 491-506.
 - Applying neural networks to satellite imagery in order to predict economic growth.

Optional reading

- Liu, Liu, Daria Dzyabura, and Natalie Mizik. “Visual listening in: Extracting brand image portrayed on social media.” *Marketing Science* 39, no. 4 (2020): 669-686.
 - Examines brand image and how reflective profiles are of the brands
- Obaid, Khaled, and Kuntara Pukthuanthong. “A picture is worth a thousand words: Measuring investor sentiment by combining machine learning and photos from news.” *Journal of Financial Economics* (2021).
 - Examines “Photo Pessimism,” and compares visual pessimistic sentiment against textual pessimistic sentiment

R coding resources

- Just like with Python, we can set up neural networks using [Keras](#) and run them with [TensorFlow](#) as the backend. This can be sufficient for image processing in R as well.

Python coding resources

- [Keras](#) is a good interface to use to set up neural networks, and [TensorFlow](#) is a good backend
- [pytorch](#) is also pretty good at working with images, with a lot of open source neural networks available
- [Caffe](#) is a neural network backend focused mostly on image processing
- For simpler operations, [scikit-image](#) can be useful
- [tensorflow_hub](#) has some pretrained image processing models.

Removed topic (2024): Individual and group traits from text

Varied

Suggested reading

- Eichstaedt, Johannes C., Hansen Andrew Schwartz, Margaret L. Kern, Gregory Park, Darwin R. Labarthe, Raina M. Merchant, Sneha Jha et al. “Psychological language on Twitter predicts county-level heart disease mortality.” *Psychological science* 26, no. 2 (2015): 159-169.
 - Psychology + Social Media = Healthcare predictions.
- Majumder, Navonil, Soujanya Poria, Alexander Gelbukh, and Erik Cambria. “Deep learning-based document modeling for personality detection from text.” *IEEE Intelligent Systems* 32, no. 2 (2017): 74-79.
 - Personality prediction using essay writing.

Additional reading

- De Choudhury, Munmun, Michael Gamon, Scott Counts, and Eric Horvitz. “Predicting depression via social media.” In *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 7, no. 1. 2013.
 - Depression prediction using social media.
- Mairesse, François, Marilyn A. Walker, Matthias R. Mehl, and Roger K. Moore. “Using linguistic cues for the automatic recognition of personality in conversation and text.” *Journal of artificial intelligence research* 30 (2007): 457-500.
 - The original paper examining personality prediction from writing – this is the Mairesse benchmark in the Majumder et al. 2017 paper.
- Colnerič, Niko, and Janez Demšar. “Emotion recognition on twitter: Comparative study and training a unison model.” *IEEE transactions on affective computing* 11, no. 3 (2018): 433-446.
 - This paper examines emotion recognition on Twitter. The final model from this paper is publicly available to use.

R coding resources

N/A

Python coding resources

- For personality detection, [SenticNet/personality-detection](#) should be reasonably accurate as far as this type of analysis goes, but note that the code is outdated, needing to be run on old versions of Theano with python 2
- [Personality Recognizer \(Mairesse\)](#) ([Github mirror of this](#)), while older, is quite usable.
- For Twitter-specific analysis, [Twitter Emotion Recognition](#) works well, though again it is based on Theano (which itself is deprecated)