

Machine Learning for Social Science (2021 Fall)

Dr. Richard M. Crowley

Version 2021.08.26

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 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 part because different applicable methodologies are more commonly used (or better used) in different disciplines. As such, required readings are pulled from Accounting, Computer Science, Economics, 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 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-requisite

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.

¹Homeworks and the final project can be completed in other languages, but any in class examples will be done in python 3 and R. 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.

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. Subject to agreement by Datacamp, Datacamp access will be provided for free at the start of the semester to all enrolled students.

Presentations

To facilitate discussion in each class session, two papers (usually the least technical paper(s)) will 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.

Final project

To be announced around Session 8 of the course.

Assignments

Throughout the course, there will be short assignments to illustrate how different algorithms work. These will require you to apply the methodology to a small dataset using your preferred programming language (python or R is recommended). Note that 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 groups.

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 an [Rmarkdown file](#). 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.
4. Make separate files for code and for results and explanations, though I would encourage you to explore the above methods first.

Grading

The following weights apply for course grading:

Class presentations	20%
Pre-class summaries and participation	20%
Assignments (equally weighted)	30%
Final Project	30%
<hr/> Total	<hr/> 100%

Topics

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.

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.
- 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. 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: This paper is quite long and includes a rather complex method they call CDDF – we will cover this method in Session 8. As such, just focus on the more straightforward benefit of using XGBoost in such a setting rather than the benefits derived from CDDF.

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.

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 Kapons, Steven J. Monahan, Harm H. Schütt, and Eric H. Weisbrod. “Forecasting Earnings Using k-Nearest Neighbor Matching.” Available at SSRN (2020).
 - Analyst forecasts using KNN.
2. Qiu, Yue, Tian Xie, and Y. U. Jun. “Forecast combinations in machine learning.” (2020)
 - A straightforward use of Ensemble methods in economics.

Optional reading

- 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.
- More TBA

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.

Text data

Week 4: Basic natural language processing (NLP) and sentiment

Sentiment dictionaries · Supervised classifiers

Required reading

- 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.
- 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.
- 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

- Loughran, Tim, and Bill McDonald. “Measuring firm complexity.” Available at SSRN 3645372 (2020).
 - 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 Alexander, Stephan Hollander, Laurence Van Lent, and Ahmed Tahoun. Firm-level exposure to epidemic diseases: Covid-19, SARS, and H1N1. No. w26971. National Bureau of Economic Research, 2020.
 - 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 5: Linguistics

Grammatical structure · Named-Entity Recognition (NER)

Required reading

1. 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.
2. 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.

Optional reading

- TBA

R coding resources

- For grammar parsing, consider [spacyr](#) or [coreNLP](#).

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.

Week 6: Word embeddings and topic modeling

LDA · Word2vec · Universal Sentence Encoder (USE)

Required reading

1. Huang, Allen H., Reuven Lehav, 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.
2. Crowley, Richard M., Wenli Huang, and Hai Lu. “Executive Tweets.” Rotman School of Management Working Paper (2020).
 - An application of USE to compare the meaning of tweets. Also uses a variant of LDA for tweet classification: Twitter-LDA.
3. 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.

Optional reading

- 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.
- Cer, Daniel, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant et al. “Universal sentence encoder.” arXiv preprint arXiv:1803.11175 (2018).
 - Technical details for USE.
- 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.
- 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.
 - Another nice use of LDA: predicting corporate misreporting based on annual report text.

R coding resources

- For word embeddings, you can use [word2vec](#)
- [stm](#) (Structural Topic Models) is a nice twist on LDA. [lda](#), [topicmodels](#), and [RMallet](#) also work.
- [TensorFlow](#) is also available in R.

Python coding resources

- For LDA and word2vec, [gensim](#) is both easy to use and fast to run.
 - Note: It has some incompatibilities with Python 3.9 (as of July 2021).
- [TensorFlow](#) has embedding implementations available – harder to implement than those of gensim, but TensorFlow includes more modern approaches and should usually run faster.

Week 7: Individual traits from text

Varied

Required reading

1. 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.
2. 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.
3. 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.

Optional reading

- 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 (None that I am aware of currently)

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)

Machine learning and econometrics

Week 8: Causal ML

Double/debiased/neyman ML · Custom estimators

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. 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

- 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.

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 9: Policy prediction

Biased estimators

Required reading

- Kleinberg, Jon, Jens Ludwig, Sendhil Mullainathan, and Ziad Obermeyer. “Prediction policy problems.” *American Economic Review* 105, no. 5 (2015): 491-95.
 - An overview of when prediction is more important than causation in policy problems, along with an explanation and example of how to solve such problems.
- 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 of last week’s material.
- 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

- 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.

R coding resources

Python coding resources

Week 10: Bias

SHAP

Required reading

- 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.
- Lundberg, Scott M., Bala Nair, Monica S. Vavilala, Mayumi Horibe, Michael J. Eisses, Trevor Adams, David E. Liston et al. “Explainable machine-learning predictions for the prevention of hypoxaemia during surgery.” Nature biomedical engineering 2, no. 10 (2018): 749-760.
 - A practical use of SHAP for model explainability.

Optional reading

- Lundberg, Scott, and Su-In Lee. “A unified approach to interpreting model predictions.” arXiv preprint arXiv:1705.07874 (2017).
 - The technical details behind SHAP
- 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.

Neural networks

Neural networks are useful in a lot of contexts, providing more flexibility and a variety of model structures that differ from what we have seen so far.

Week 11: Neural networks and transfer learning

Neural networks, broadly · Reinforcement learning · Transfer learning

Required reading

- Huang, Allen, Hui Wang, and Yi Yang. “The Informativeness of Text, the Deep Learning Approach.” (2020).
 - Applies BERT with a bit of transfer learning.
- Liu, Xiao, Dokyun Lee, and Kannan Srinivasan. “Large-scale cross-category analysis of consumer review content on sales conversion leveraging deep learning.” *Journal of Marketing Research* 56, no. 6 (2019): 918-943.
 - Uses a CNN to understand when and why reviews online impact consumer behavior.

Optional reading

R coding resources

- Just like with Python, we can set up neural networks using [Keras](#) and run them with [TensorFlow](#) as the backend.
- For simpler neural networks, [nnet](#) can be sufficient

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
- [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](#).

Week 12: Image classification

Convolutional neural networks

Required reading

- Marketing research using image processing: 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
- Zhang, Shunyuan, Dokyun DK Lee, Param Vir Singh, and Kannan Srinivasan. “How much is an image worth? Airbnb property demand estimation leveraging large scale image analytics.” *Airbnb Property Demand Estimation Leveraging Large Scale Image Analytics* (May 25, 2017) (2017).
 - Examines how images in listings impact AirBNB properties

Optional reading TBD

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
- [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.