**Required**: Kevin Murphy, *Machine Learning, a Probabilistic Perspective*, 1st edition, MIT Press (2012) [author disavows the eBook version sold on Amazon, only buy an eBook from MIT Press]

**Supplemental:** Christopher Bishop, *Pattern Recognition and Machine Learning,* Springer (2011 printing)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Weekly Schedule** | | | | |
| **Week** | **Topics** | **Learning Goal** | **Evidence Required to Demonstrate Goals Are Achieved.**  **Be precise!!** | **Learning Activities That Will Produce the Evidence (quizzes, coding homework, etc.)**  **Be precise!! If it is a quiz, give some sample questions. If it is a coding assignment, give the specifications for the inputs, outputs, and algorithms to be implemented.** |
| 1 | Overview of Machine Learning: history, relation to classical statistics **Text:** Chapter 1; | 1. Explain the idea of intelligence especially as it relates to computers.  2. Explain what it means for a machine to “learn”.  3. Explain common learning paradigms  4. Be able to formulate a well-posed learning problem for a real-world task by identifying the task, performance measure, and training experience  5. Be able to identify Machine Learning in real-world scenarios | Interact and contribute to in-class group activities and discussions  Articulate answers for (in-class) questions:  What is intelligence and what is the difference of human intelligence versus computer intelligence?  What Does it Mean to Learn? | Video about how to use Machine Learning in Real World Applications  Play the intelligent paper game  Ask students to give other examples of supervised and unsupervised learning situations (in class discussion)  Ask students to formulate a learning problem (in class discussion)  Ask students to give other applications of Machine Learning (in class discussion) |
| 2 | **Linear Regression:** model specification issues, maximum likelihood estimation, ridge regression **Text:** Chapter 7, except 7.4 and 7.6 | 1.  To know simple linear regression, intercept slope coefficient parameter, least squares, sum of squares, population regression line, bias and unbiased, standard error, residual standard error  2. Design k-NN Regression and Decision Tree Regression  3. To Understand how to estimate the regression coefficients  4. Implement learning for Linear Regression using three optimization techniques: (1) closed form, (2) gradient descent, (3) stochastic gradient descent  5. To be able to choose a Linear Regression optimization technique that is appropriate for a particular dataset by analyzing the tradeoff of computational complexity vs. convergence speed | Explain the difference between a deterministic vs stochastic vs regression models  Explain the interpretation of the beta estimates  Explain difference between bias and variance?  Find the MLE for the mean of normally distributed variables (intro example) and (more applied) MLE for regression beta coefficient.  - Explain when to use closed form vs numerical solutions? (closed form is always the preferred way, but of course it almost never exists)  - Explain difference between train and test error.  - Penalized regression methods (lasso, ridge) | Application cards. Record two or three different ways to apply regression to a real-world situation.  Calculate the beta for IBM stock using webscrapping: In finance they often refer to the beta of a stock: that is nothing else than the  estimator of a regression run on the past 60months of a security return as a function of market return ( the S&P 500 index or the CRSP index).(for example, IBM closing monthly prices can be scraped from here, they can be downloaded as well, but scraping should be used for full points. <https://finance.yahoo.com/quote/IBM/history?period1=1380772800&period2=1538539200&interval=1mo&filter=history&frequency=1mo> Students will be able to pull in the prices, but they will need to calculate returns = (price at current month – price at previous month)/price at previous month. Similarly, S&P500 data is available here [https://finance.yahoo.com/quote/%5EGSPC/history?p=^GSPC&.tsrc=fin-srch](https://finance.yahoo.com/quote/%5EGSPC/history?p=%5eGSPC&.tsrc=fin-srch))   * Expected Learning Outcomes Statements (end of class survey) |
| 3 | **Logistic Regression:** model fitting methods (steepest descent, Newton’s, L1 and L2 regularization, etc.) **Online learning algorithms**: structured prediction, regret minimization, stochastic optimization, risk minimization, LMS algorithm, perceptron algorithm, relation to Bayesian techniques. **Text:** Chapter 8.1-8.3.6, 8.5, 13.3, | Apply the principle of maximum likelihood estimation (MLE) to learn the parameters of a probabilistic model  Given a discriminative probabilistic model, derive the conditional log-likelihood, its gradient, and the corresponding Bayes Classifier  Explain the practical reasons why we work with the **log** of the likelihood  Implement logistic regression for binary or multiclass classification  Prove that the decision boundary of binary logistic regression is linear  For linear regression, show that the parameters which minimize squared error are equivalent to those that maximize conditional likelihood | What situations require logistic vs linear? Explain how applying a linear model to a binary response would fail.  ROC Charts. Construct one by hand.  How do you interpret the odds ratios of logistic regression?  Explain the difference and similarities between stochastic gradient descent and gradient descent.  What is LMS? Comment on the relationship to stochastic gradient descent. | Implement a spam detection logistic regression model. Use crossvalidation to compare models.    “Muddiest Point” Activity. conducted at the end of the week. Students are anonymously asked to report on a piece of paper what idea about regression was confusing or unclear.   * *Use a Naïve Bayes model to predict Sports vs Non Sports messages.* * *Non-graded quiz.* |
| 4 | **Support Vector Machines:** uses in regression and classification, optimization, choice of parameter C, probabilistic interpretation **Kernel Methods:** RBF kernels, Mercer kernels, Matern kernels, Linear kernels, String kernels, Kernels derived from probabilistic generative models; the “kernel trick” and its uses: nearest neighbor classification, K-medoids clustering, ridge regression, and PCA **Text:** Chapter 14.1, 14.2, 14.5, supplemental notes by Ng | Motivate the learning of a decision boundary with large margin  Compare the decision boundary learned by SVM with that of Perceptron  Derive the hard-margin SVM primal formulation  Derive the Lagrangian dual for a hard-margin SVM  Describe the mathematical properties of support vectors and provide an intuitive explanation of their role  Employ slack variables to obtain the soft-margin SVM  Employ the kernel trick in common learning algorithms  Use the "kernel trick" to obtain a computational complexity advantage over explicit feature transformation  Implement the K-Means algorithm  Connect the nonconvexity of the K-Means objective function with the (possibly) poor performance of random initialization  PCA and Dimensionality Reduction | * What is a hyperplane? * Explain the Maximal Margin Classifier. * Perfectly separable vs non-separable. * Explain Support Vector Classifier vs Support Vector Machine * Give simple example of how a polynomial kernel solves a common non-linear example: | Think-pair-share. Having students turn to the person next to them, discuss the question, and then vote again  **Think-pair-share. A**llow a couple of minutes for each individual student to thinkit through. Next, each student turns to the student next to him/her to discuss the question/answer as a pair. ask student pairs to sharetheir response with the class.  **One Minute Paper**.List your major questions related to Support Vector Machines.  Implement several SVM models for spam detection, use the same data used for the logistic regression example  - Compare several models using ROC charts and crossvalidation  - use PCA and then svm/regression, different kernels form svm, compare models |
| 5 | **Decision Trees:** basis in information theory, when appropriate to use, growing and pruning, detecting and avoiding over-fitting; pros and cons with respect to other ML techniques; random forests; ID3 and related algorithms; use of bias; training with incomplete data **Boosting:** why it works so well, specific types of boosting, ensemble learning **Text:** Chapters 2.8, 16.2, 16.4, 16.6 , supplemental notes by Mitchell, Yoav & Schapire | 1. Implement Decision Tree training and prediction  2. Use effective splitting criteria for Decision Trees and be able to define entropy, conditional entropy, and mutual information / information gain  3. Describe the inductive bias of a decision tree  4. Judge whether a decision tree is "underfitting" or "overfitting"  5. Explain the difference between true error and training error  6. Implement a pruning or early stopping method to combat overfitting in Decision Tree learning  7. Identify when a model is overfitting  8. Add a regularizer to an existing objective in order to combat overfitting | - Interpretability. How are trees appealing?  - Types of impurity measures? What are the advanatages/disadvantages of misclassification, Gini, cross-entropy?  - Explain how things simplify with a categorical predictor.  - When is a loss matrix useful?  - Missing predictor values?  - Comment on the instability of trees.  - Comment on lack of smoothness. | **Student Generated Exam questions:** will the students be able to identify the key strengths and weaknesses of the decision tree models.  How a decision tree classifier can be used to distinguish between accesses by human users and those by Web robots. The input data was obtained from a Web server log.  Reading:  [Emerging Artificial Societies Through Learning](http://jasss.soc.surrey.ac.uk/9/2/9.html) |
| 6 | **Deep Learning:** background in feed-forward neural networks (multilayer perceptrons) and Restricted Boltzmann machines; deep generative models (sigmoid networks, Boltzmann machines, belief networks) **Text:** Chapters 16.5, 27.7, 28.1-28.2 | Combine simpler models (e.g. linear regression, binary logistic regression, multinomial logistic regression) as components to build up feed-forward neural network architectures  Explain the reasons why a neural network can model nonlinear decision boundaries for classification  Identify (some of) the options available when designing the architecture of a neural network  Implement a feed-forward neural network | * Discussion: Bayesian Neural Nets and the NIPS 2003 Challenge: comparison between Bayesian neural nets, boosted trees, boosted neural nets, bagged neural nets | Video. Amazing applications of Deep Learning  3Blue1Brown Youtube series on Deep Learning.  Build a simple neural network using tensorflow.  One Minute Paper:What you believe to be the most important/significant concept from Deep Learning. |
| 7 | **Deep Learning (continued):** training of deep networks, applications of deep networks **Midterm Review** **Text:** Chapters 28.3-28.4 | Construct a computation graph for a function as specified by an algorithm  Carry out the backpropagation on an arbitrary computation graph  Construct a computation graph for a neural network, identifying all the given and intermediate quantities that are relevant  Instantiate the backpropagation algorithm for a neural network  Instantiate an optimization method (e.g. Stochastic Gradient Descent (SGD)) and a regularizer (e.g. L2) when the parameters of a model are comprised of several matrices corresponding to different layers of a neural network  Apply the empirical risk minimization framework to learn a neural network  Use the finite difference method to evaluate the gradient of a function  Identify when the gradient of a function can be computed at all and when it can be computed efficiently |  | Student-Generated Exam Questions. |
| 8 | **Midterm Exam Clustering:** measures of dissimilarity; k-means clustering, clustering in mixture models **Text:** Chapters 25.1, 11.1-11.3 | Distinguish between coordinate descent and block coordinate descent  Define an objective function that gives rise to a "good" clustering  Apply block coordinate descent to an objective function preferring each point to be close to its nearest objective function to obtain the K-Means algorithm  Implement the K-Means algorithm  Connect the nonconvexity of the K-Means objective function with the (possibly) poor performance of random initialization | **MIDTERM REVIEW** | **MIDTERM REVIEW** |
| 9 | **Expectation Maximization:** basic idea, Jensen’s inequality, the EM algorithm, EM applied to various distributions (mixture of Gaussians), theoretical basis **Text:** Chapters 11.4, supplemental notes by Ng | Explain the relationship between parameters and hidden variables.  Construct generative stories for clustering and dimensionality reduction.  • Draw a graph explaining how EM works by constructing convex lower bounds.  • Implement EM for clustering with mixtures of Gaussians, and contrasting it with k-means.  • Evaluate the differences between EM and gradient descent for hidden variable models. | Aside from the fact that GMMs (Gaussian Mixture Models) use soft assignments and K-means uses hard assignments, there are other differences between the two approaches. What are they? | Homework. Textbook problems 11.1 to 11.15  Prove Jensen’s inequality using the definition of concavity and induction. |
| 10 | **Dimensionality Reduction:** classical Principal Components Analysis, PCA theorem; Singular Value Decomposition, Probabilistic PCA, EM algorithm for PCA, PCA for categorical data, PCA for paired and multi-view data **Text:** Chapters 12.2, 12.4, 12.5 | Define the sample mean, sample variance, and sample covariance of a vector-valued dataset  Identify examples of high dimensional data and common use cases for dimensionality reduction  Draw the principal components of a given toy dataset  Given a set of principal components, project from high to low dimensional space and do the reverse to produce a reconstruction  Explain the connection between PCA, eigenvectors, eigenvalues, and covariance matrix  Use common methods in linear algebra to obtain the principal components |  |  |
| 11 | **Graphical Techniques:** Bayesian nets, Directed Gaussian Models (DGM), plate notation, d-separation, Markov random fields, Hammersley-Clifford theorem and its effects, training maxent models, pseudo-likelihood, stochastic maximum likelihood learning **Text:** Chapter 10, 19.1-19.5 | Understand basic principles and techniques of probabilistic graphical models  Create suitable models for any given problem  Derive the algorithm (equations, data structures etc) needed to apply the model to data |  |  |
| 12 | **Graphical Techniques (continued):** belief propagation (serial, parallel, Gaussian, and other variants), variable elimination, junction trees, intractability of exact inference, approximate inference methods, max sum and max product **Text:** Chapter 20 | Describe the elements of a graphical model learning  algorithm, and the main methods for each  • Given a Beta or Dirichlet prior distribution and a sample of cases, compute the posterior distribution for a local distribution for a Bayesian network  • Compute the relative posterior probability for two structures for a Bayesian network when the prior distribution is a Beta or Dirichlet mixture distribution  • Describe methods for learning graphical models with missing observations and hidden variables  Demonstrate your skills by doing a reasonably challenging project |  | Homework. Textbook problems 19.1 to 19.3, 20.1 to 20.4  One Minute Paper: What you believe to be the most important/significant concept from Graphical Techniques. |
| 13 | **Structured Prediction:** Markov and Hidden Markov (HMM) models, inference algorithms (forward, Viterbi, backward filtering, etc.), training methods, Baum-Welch algorithm, model selection, Conditional Random Fields (CRF), label-bias problem, uses in handwriting recognition, protein analysis, and stereopsis **Text:** Chapters 17.1-17.5, 19.6-19.7, CRF tutorial paper by Sutton & McCallum | 1. Show that structured prediction problems yield high-computation inference problems  2. Define the three key problems for an HMM: evaluation, decoding, and marginal computation  3. Compare the reinforcement learning paradigm to other learning paradigms  4. Cast a real-world problem as a Markov Decision Process  5. Derive a dynamic programming algorithm for computing the marginal probabilities of an HMM  6. Interpret the forward-backward algorithm as a message passing algorithm  7. Implement supervised learning for an HMM  8. Implement the forward-backward algorithm for an HMM  Implement the Viterbi algorithm for an HMM  9. Implement a minimum Bayes risk decoder with Hamming loss for an HMM | How Hard Is Inference for Structured Prediction? | Particle Swarm Optimization |

**Course Overview**

Machine learning techniques enable us to automatically extract features from data so as to solve predictive tasks, such as speech recognition, object recognition, machine translation, question-answering, anomaly detection, medical diagnosis and prognosis, automatic algorithm configuration, personalisation, robot control, time series forecasting, and much more. Learning systems adapt so that they can solve new tasks, related to previously encountered tasks, more efficiently.

This course will introduce the field of Machine Learning, in particular focusing on the core concepts of supervised and unsupervised learning. In supervised learning we will discuss algorithms which are trained on input data labelled with a desired output, for instance an image of a face and the name of the person whose face it is, and learn a function mapping from the input to the output. Unsupervised learning aims to discover latent structure in an input signal where no output labels are available, an example of which is grouping web-pages based on the topics they discuss. Students will learn the algorithms which underpin many popular Machine Learning techniques, as well as developing an understanding of the theoretical relationships between these algorithms. The practicals will concern the application of machine learning to a range of real-world problems.

**Learning outcomes**

On completion of the course students will be expected to:

* Have a good understanding of the fundamental issues and challenges of machine learning: data, model selection, model complexity, etc.
* Have an understanding of the strengths and weaknesses of many popular machine learning approaches.
* Appreciate the underlying mathematical relationships within and across Machine Learning algorithms and the paradigms of supervised and un-supervised learning.
* Be able to design and implement various machine learning algorithms in a range of real-world applications.

(EXPANDED) LEARNING OUTCOMES

* Defines and is able to explain basic concepts in machine learning (e.g. training data, feature, model selection, loss function, training error, test error, overfitting)
* Recognizes various machine learning problems and methods suitable for them: supervised vs unsupervised learning, discriminative vs generative learning paradigm, symbolic vs numeric data
* Knows the basics of a programming environment (such as R or python/numpy/scipy) suitable for machine learning applications
* Is able to implement at least one distance-based, one linear, and one generative classification method, and apply these to solving simple classification problems
* Is able to implement and apply linear regression to solve simple regression problems
* Explains the assumptions behind the machine learning methods presented in the course
* Implements testing and cross- validation methods, and is able to apply them to evaluate the performance of machine learning methods and to perform model selection
* Comprehends the most important clustering formalisms (distance measures, k-means clustering, hierarchical clustering)
* Explains the idea of the k-means clustering algorithm and is able to implement it
* Is able to implement a method for hierarchical clustering and can interpret its results

Example of a Dataset Analysis

# Ames Housing Dataset

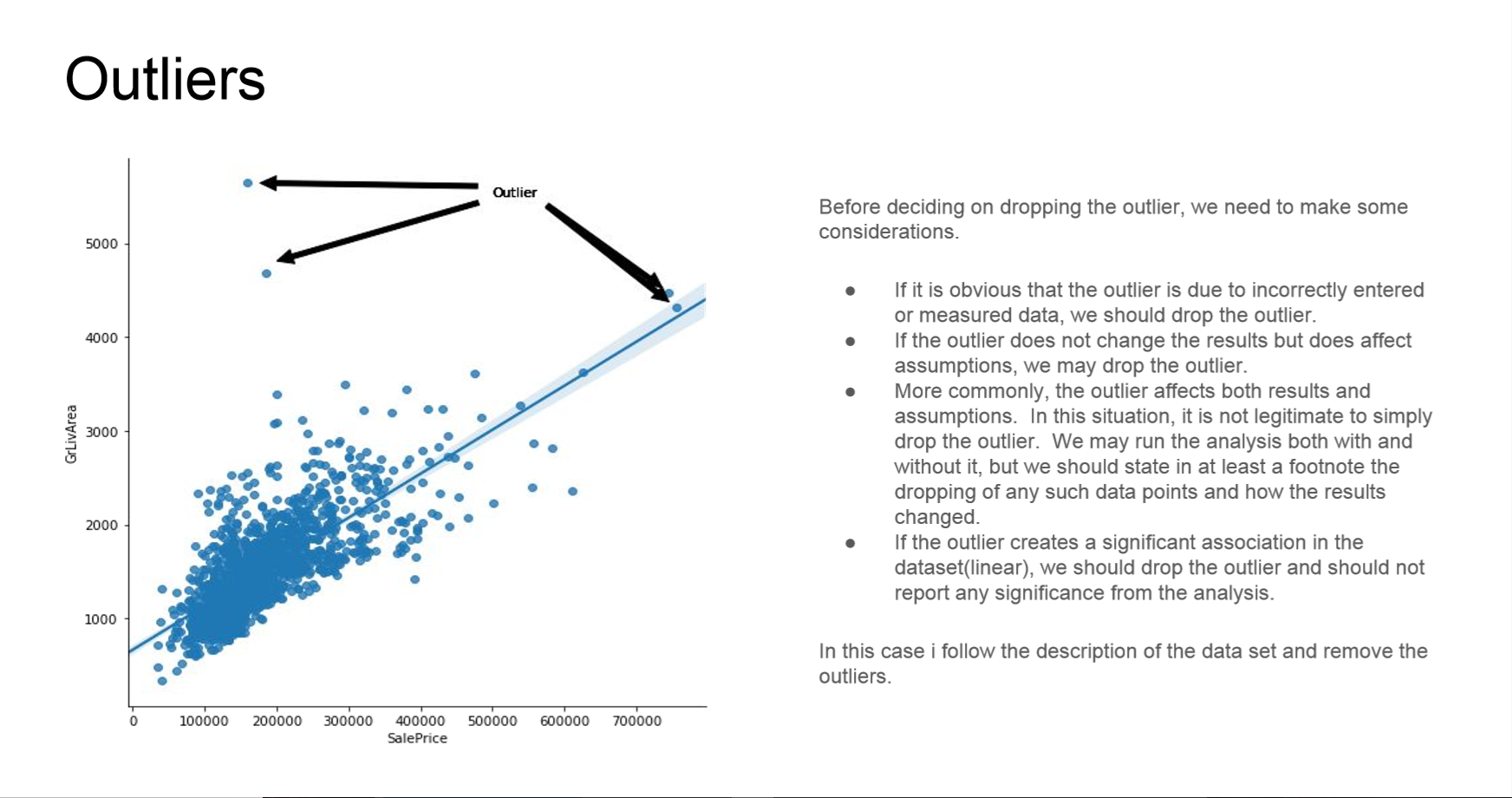
# Abstract

Ames Housing Dataset contained what a typical home buyer would want to know before making a purchase. In this analysis i will try answer question such as:

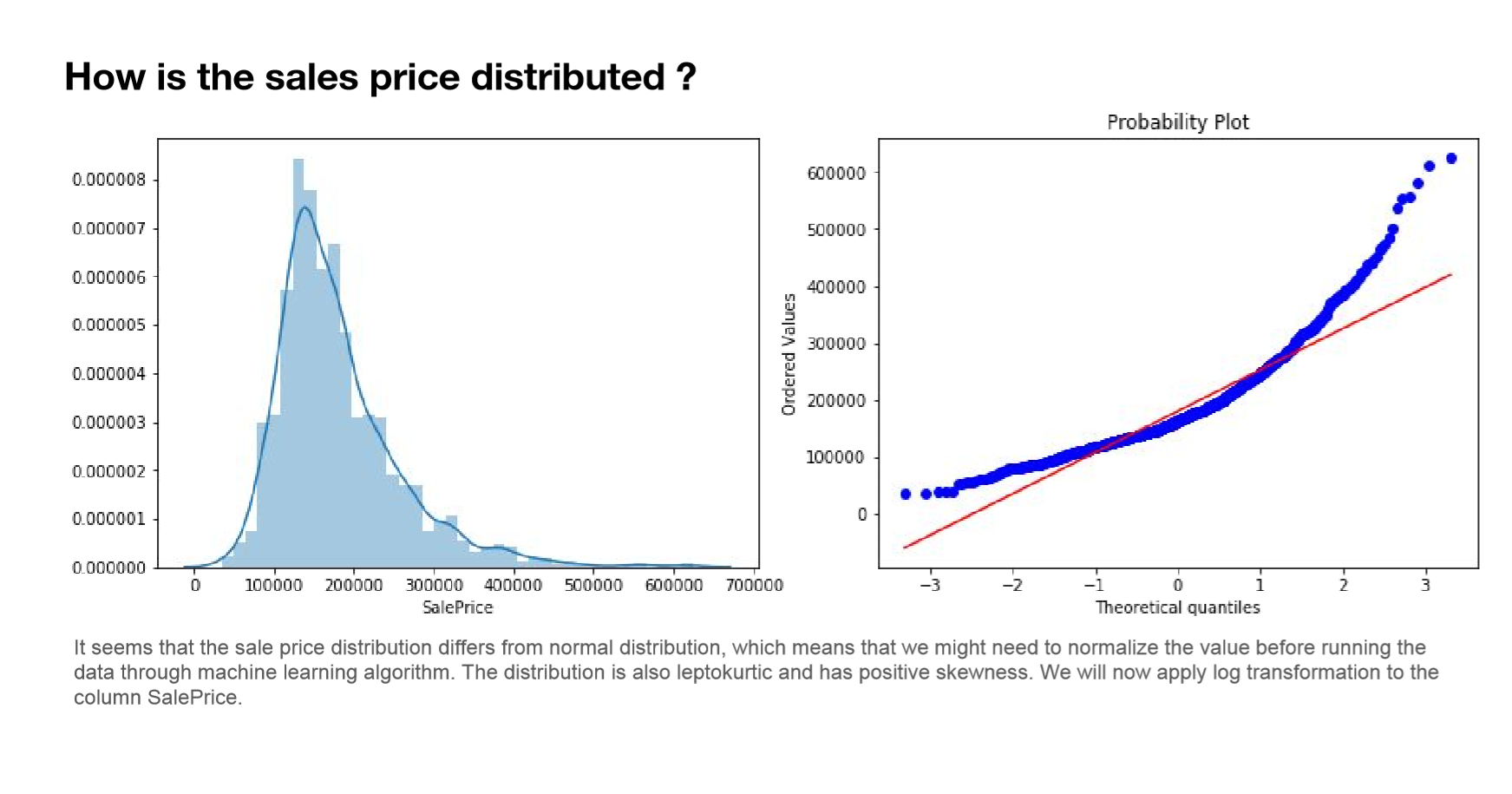
* What are the variables that have the most inﬂuences on the sales price ?
* Do Overall Quality of the house affects the sale price ?
* Do Sales Condition of the house affects the sale price ?

The relevant variables are plotted using box plot,scatter plot and bar chart to find connection between them. Some statistical test such as correlation and multicollinearity also have been done. We found out that variable such as age of the house and size of the house play a major part in driving up the sales price.

# Treatment of Outliers

[](https://github.com/faizalazman/Ames-Housing-Dataset/blob/master/Outliers.png)

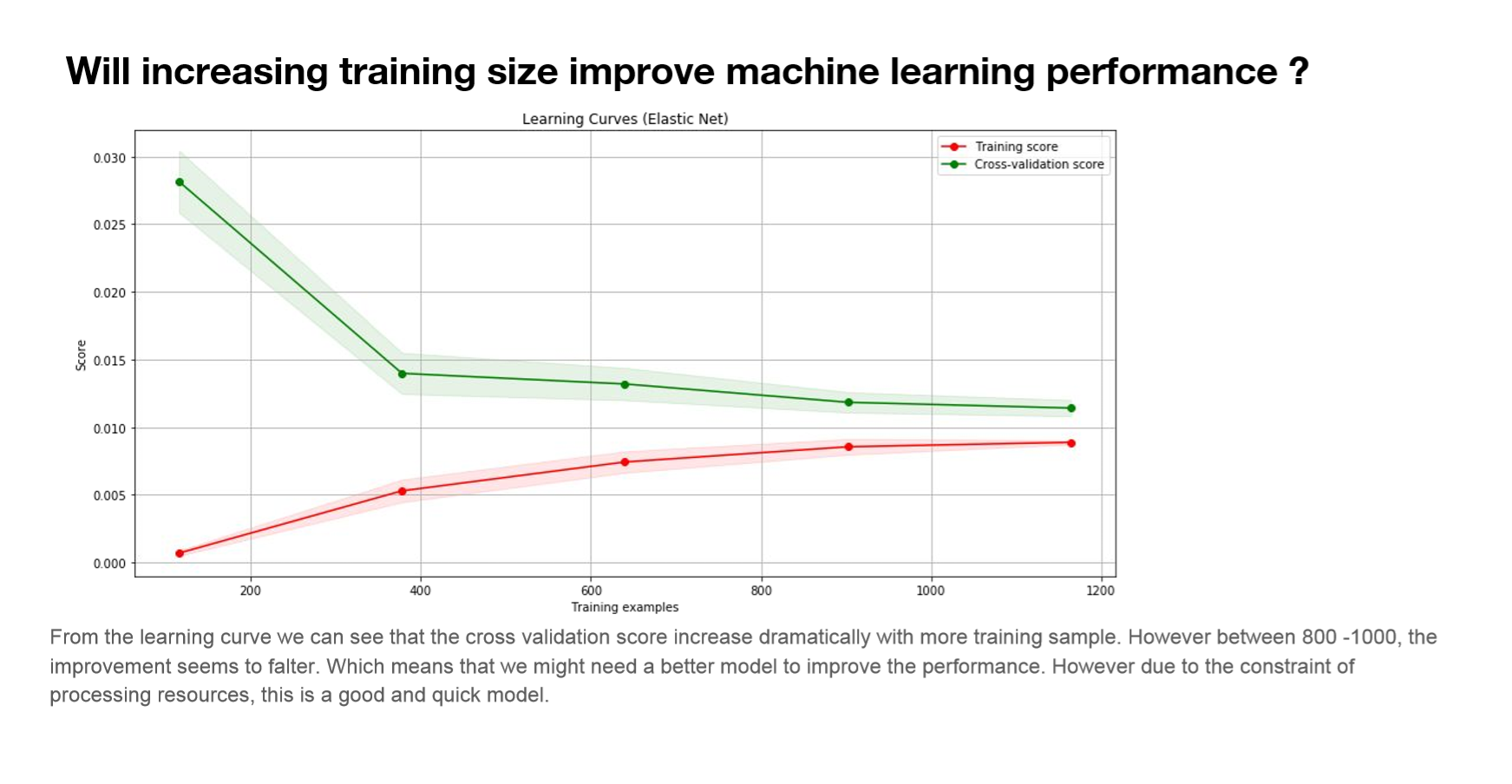
# Distribution of Sales Price

[](https://github.com/faizalazman/Ames-Housing-Dataset/blob/master/Sale%20Price%20B.png)

# What affects House Price?

[](https://github.com/faizalazman/Ames-Housing-Dataset/blob/master/Sale%20conds.png)

# Machine Learning Performance

[](https://github.com/faizalazman/Ames-Housing-Dataset/blob/master/ML%20Curves.png)

# Conclusions

What are the variables that have the most inﬂuences on the sales price ?

* Above ground living area square feet,size of garage and total square feet of basement area.

How is the sales price distributed ?

* The sale price doesn’t follow normal distribution. Fixed using natural log.

Does the variable shows multicollinearity ?

* Yep.

Do Overall Quality of the house affects the sale price ?

* Higher rating for overall quality increase the sale price.

Do Sales Condition of the house affects the sale price

* The price of normal house has more spread than any of the other type of sale condition. Sales within family on the other hand, shows a small spread which makes sense because people tend to value their family more and decide to not charge exorbitant price. Partial home which is new house has higher mean price value than other type of sales condition. Therefore, i do think sale condition has low impact on the sales price except for new houses.

What is the best machine learning algorithm for this dataset. Ensemble or simple model such as Lasso and Elastic Net ?

* Elastic Net does the job just fine. Huge number of features makes linear models more powerful. The performance gain using ensemble method is not worth it considering the computation resources required. Elastic Net managed to score 0.1094 on the training set while scoring 0.1181 on the training set.

Example of Assignment

## Assignment Understanding and Predicting Property Maintenance Fines

This assignment is based on a data challenge from the Michigan Data Science Team ([MDST](http://midas.umich.edu/mdst/)).

The Michigan Data Science Team ([MDST](http://midas.umich.edu/mdst/)) and the Michigan Student Symposium for Interdisciplinary Statistical Sciences ([MSSISS](https://sites.lsa.umich.edu/mssiss/)) have partnered with the City of Detroit to help solve one of the most pressing problems facing Detroit - blight. [Blight violations](http://www.detroitmi.gov/How-Do-I/Report/Blight-Complaint-FAQs) are issued by the city to individuals who allow their properties to remain in a deteriorated condition. Every year, the city of Detroit issues millions of dollars in fines to residents and every year, many of these fines remain unpaid. Enforcing unpaid blight fines is a costly and tedious process, so the city wants to know: how can we increase blight ticket compliance?

The first step in answering this question is understanding when and why a resident might fail to comply with a blight ticket. This is where predictive modeling comes in. For this assignment, your task is to predict whether a given blight ticket will be paid on time.

All data for this assignment has been provided to us through the [Detroit Open Data Portal](https://data.detroitmi.gov/). **Only the data already included in your Coursera directory can be used for training the model for this assignment.** Nonetheless, we encourage you to look into data from other Detroit datasets to help inform feature creation and model selection. We recommend taking a look at the following related datasets:

* [Building Permits](https://data.detroitmi.gov/Property-Parcels/Building-Permits/xw2a-a7tf)
* [Trades Permits](https://data.detroitmi.gov/Property-Parcels/Trades-Permits/635b-dsgv)
* [Improve Detroit: Submitted Issues](https://data.detroitmi.gov/Government/Improve-Detroit-Submitted-Issues/fwz3-w3yn)
* [DPD: Citizen Complaints](https://data.detroitmi.gov/Public-Safety/DPD-Citizen-Complaints-2016/kahe-efs3)
* [Parcel Map](https://data.detroitmi.gov/Property-Parcels/Parcel-Map/fxkw-udwf)

We provide you with two data files for use in training and validating your models: train.csv and test.csv. Each row in these two files corresponds to a single blight ticket, and includes information about when, why, and to whom each ticket was issued. The target variable is compliance, which is True if the ticket was paid early, on time, or within one month of the hearing data, False if the ticket was paid after the hearing date or not at all, and Null if the violator was found not responsible. Compliance, as well as a handful of other variables that will not be available at test-time, are only included in train.csv.

Note: All tickets where the violators were found not responsible are not considered during evaluation. They are included in the training set as an additional source of data for visualization, and to enable unsupervised and semi-supervised approaches. However, they are not included in the test set.

**File descriptions** (Use only this data for training your model!)

train.csv - the training set (all tickets issued 2004-2011)

test.csv - the test set (all tickets issued 2012-2016)

addresses.csv & latlons.csv - mapping from ticket id to addresses, and from addresses to lat/lon coordinates.

Note: misspelled addresses may be incorrectly geolocated.

**Data fields**

train.csv & test.csv

ticket\_id - unique identifier for tickets

agency\_name - Agency that issued the ticket

inspector\_name - Name of inspector that issued the ticket

violator\_name - Name of the person/organization that the ticket was issued to

violation\_street\_number, violation\_street\_name, violation\_zip\_code - Address where the violation occurred

mailing\_address\_str\_number, mailing\_address\_str\_name, city, state, zip\_code, non\_us\_str\_code, country - Mailing address of the violator

ticket\_issued\_date - Date and time the ticket was issued

hearing\_date - Date and time the violator's hearing was scheduled

violation\_code, violation\_description - Type of violation

disposition - Judgment and judgement type

fine\_amount - Violation fine amount, excluding fees

admin\_fee - $20 fee assigned to responsible judgments

state\_fee - $10 fee assigned to responsible judgments late\_fee - 10% fee assigned to responsible judgments discount\_amount - discount applied, if any clean\_up\_cost - DPW clean-up or graffiti removal cost judgment\_amount - Sum of all fines and fees grafitti\_status - Flag for graffiti violations

train.csv only

payment\_amount - Amount paid, if any

payment\_date - Date payment was made, if it was received

payment\_status - Current payment status as of Feb 1 2017

balance\_due - Fines and fees still owed

collection\_status - Flag for payments in collections

compliance [target variable for prediction]

Null = Not responsible

0 = Responsible, non-compliant

1 = Responsible, compliant

compliance\_detail - More information on why each ticket was marked compliant or non-compliant

## Evaluation

Your predictions will be given as the probability that the corresponding blight ticket will be paid on time.

The evaluation metric for this assignment is the Area Under the ROC Curve (AUC).

Your grade will be based on the AUC score computed for your classifier. A model which with an AUROC of 0.7 passes this assignment, over 0.75 will recieve full points.

For this assignment, create a function that trains a model to predict blight ticket compliance in Detroit using train.csv. Using this model, return a series of length 61001 with the data being the probability that each corresponding ticket from test.csv will be paid, and the index being the ticket\_id.

Example:

ticket\_id

284932 0.531842

285362 0.401958

285361 0.105928

285338 0.018572

...

376499 0.208567

376500 0.818759

369851 0.018528

Name: compliance, dtype: float32

### Hints

* Make sure your code is working before submitting it to the autograder.
* Print out your result to see whether there is anything weird (e.g., all probabilities are the same).
* Generally the total runtime should be less than 10 mins. You should NOT use Neural Network related classifiers (e.g., MLPClassifier) in this question.
* Try to avoid global variables. If you have other functions besides blight\_model, you should move those functions inside the scope of blight\_model.
* Refer to the pinned threads in Week 4's discussion forum when there is something you could not figure it out.

In [ ]:

**import** **pandas** **as** **pd**

**import** **numpy** **as** **np**

**def** blight\_model():

**from** **datetime** **import** datetime

**import** **pandas** **as** **pd**

**import** **numpy** **as** **np**

**from** **sklearn.ensemble** **import** RandomForestClassifier

**from** **sklearn.preprocessing** **import** MinMaxScaler

**from** **sklearn.model\_selection** **import** train\_test\_split, GridSearchCV

**from** **sklearn.metrics** **import** roc\_auc\_score

**def** time\_gap(hearing\_date\_str, ticket\_issued\_date\_str):

**if** **not** hearing\_date\_str **or** type(hearing\_date\_str)!=str: **return** 73

hearing\_date = datetime.strptime(hearing\_date\_str, "%Y-%m-**%d** %H:%M:%S")

ticket\_issued\_date = datetime.strptime(ticket\_issued\_date\_str, "%Y-%m-**%d** %H:%M:%S")

gap = hearing\_date - ticket\_issued\_date

**return** gap.days

data = pd.read\_csv('train.csv',encoding = 'ISO-8859-1')

data = data[(data['compliance'] == 0) | (data['compliance'] == 1)]

data = data[data.agency\_name != 'Neighborhood City Halls']

data = data[data.agency\_name != 'Health Department']

test\_data = pd.read\_csv('test.csv')

address = pd.read\_csv('addresses.csv')

latlons = pd.read\_csv('latlons.csv')

address = address.set\_index('address').join(latlons.set\_index('address'), how='left')

data = data.set\_index('ticket\_id').join(address.set\_index('ticket\_id'))

target = data['compliance']

feature = data.drop('compliance', axis = 1)

feature['time\_gap'] = feature.apply(**lambda** row: time\_gap(row['hearing\_date'], row['ticket\_issued\_date']), axis=1)

test\_data['time\_gap'] = test\_data.apply(**lambda** row: time\_gap(row['hearing\_date'], row['ticket\_issued\_date']), axis=1)

test\_data = test\_data.set\_index('ticket\_id').join(address.set\_index('ticket\_id'))

test\_data = test\_data.drop(['violator\_name', 'zip\_code', 'country','city','state',

'inspector\_name', 'violation\_street\_number', 'violation\_street\_name',

'violation\_zip\_code', 'violation\_description',

'mailing\_address\_str\_number', 'mailing\_address\_str\_name',

'non\_us\_str\_code',

'ticket\_issued\_date', 'hearing\_date', 'grafitti\_status','disposition','violation\_code'], axis = 1)

test = list(test\_data.columns)

train = list(feature.columns)

**for** column **in** train:

**if** column **not** **in** test:

feature = feature.drop(column, axis = 1)

feature = pd.get\_dummies(feature)

test\_data = pd.get\_dummies(test\_data)

feature.lat.fillna(method='pad', inplace=**True**)

feature.lon.fillna(method='pad', inplace=**True**)

test\_data.lat.fillna(method='pad', inplace=**True**)

test\_data.lon.fillna(method='pad', inplace=**True**)

**from** **sklearn.ensemble** **import** ExtraTreesClassifier

**from** **sklearn.datasets** **import** load\_iris

**from** **sklearn.feature\_selection** **import** SelectFromModel

clf = ExtraTreesClassifier()

clf = clf.fit(feature, target)

model = SelectFromModel(clf, prefit=**True**)

feature = model.transform(feature)

test\_data = model.transform(test\_data)

sc = MinMaxScaler()

feature = sc.fit\_transform(feature)

X\_train,X\_test, y\_train, y\_test = train\_test\_split(feature, target, test\_size = 0.3, random\_state = 0)

forest = RandomForestClassifier(n\_estimators = 400, class\_weight ='balanced', max\_depth = 7)

forest.fit(X\_train,y\_train)

test\_data = sc.fit\_transform(test\_data)

compliance = forest.predict\_proba(test\_data)[:,1]

test\_df = pd.read\_csv('test.csv', encoding = "ISO-8859-1")

test\_df['compliance'] = compliance

test\_df.set\_index('ticket\_id', inplace=**True**)

**return** test\_df.compliance

In [ ]:

blight\_model()