CSE 474/574: INTRO TO MACHINE LEARNING

HOMEWORK:3

Abhinav Reddy Chintalapuri

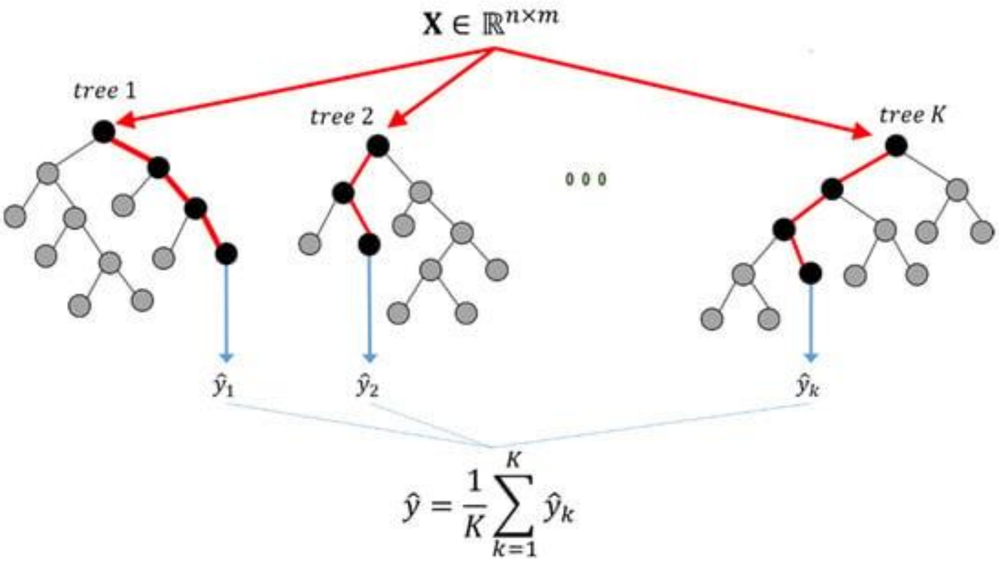
Department of Engineering Science Data Science

University at Buffalo, Buffalo, NY 14214

achintal@buffalo.edu

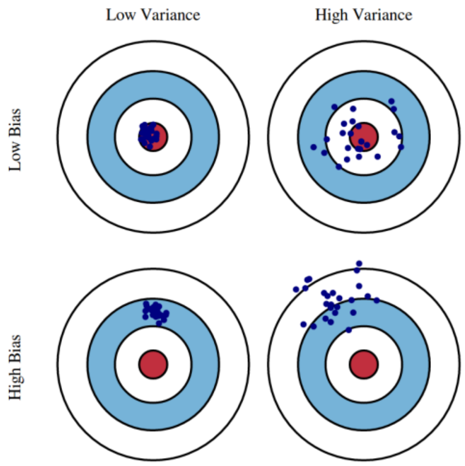
1. Random Forest:

A supervised learning algorithm called Random Forest uses a lot of Decision Trees and the ensemble learning technique. There is no interaction between Decision Trees when they are constructed because Random Forest is a Bagging technique, which means that all calculations are done concurrently. Both classification and regression issues can be resolved by RF.

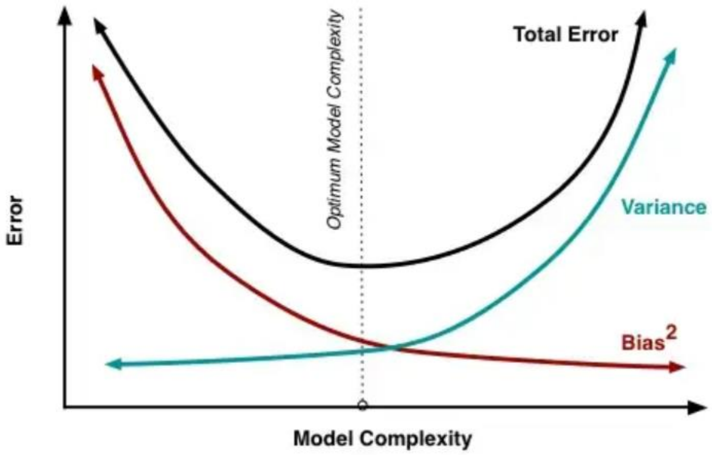

$\mathbf{X} \in \mathbb{R}^{n\times m}$
tree 1
tree 2
tree K
$\hat{y}_1$
$\hat{y}_2$
$\hat{y}_k$
$\hat{y} = \frac{1}{K}\sum_{k=1}^{K} \hat{y}_k$

Variance vs Bias:

The Mean Squared Error (MSE) of a statistical model can be expressed as the squared bias of its predictions, their variance, and the variance of some error term. Because squared bias and variance are both non-negative and capture randomness in the data, we reduce MSE by reducing the variance and bias of our model.



We call our model biased if it consistently underpredicts or overpredicts the target variable. As a result of bias in the training data, this occurs frequently in machine learning.



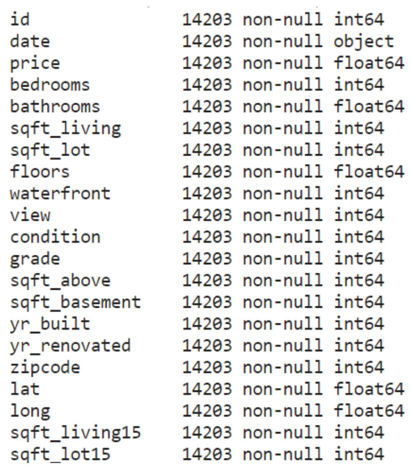
On the other hand, variation partially captures the generalizability of the model. Another way to put it is that it calculates how much our prediction would alter if we trained it using different data. High variance frequently means that we are overfitting to our training data and are finding patterns and complexity that are the result of randomness rather than a true trend. Since our model predicts the target variable more accurately when averaged over several predictions, a more complex or flexible model will typically have lower bias but higher variance due to overfitting. On the other hand, despite having lower variance, an unfit or oversimplified model will probably be more biased because it lacks the resources to fully capture data trends.

Dataset:

We must project the sales prices of homes in the county of Seattle using this dataset. Properties bought between 2014 and 2015 are included.

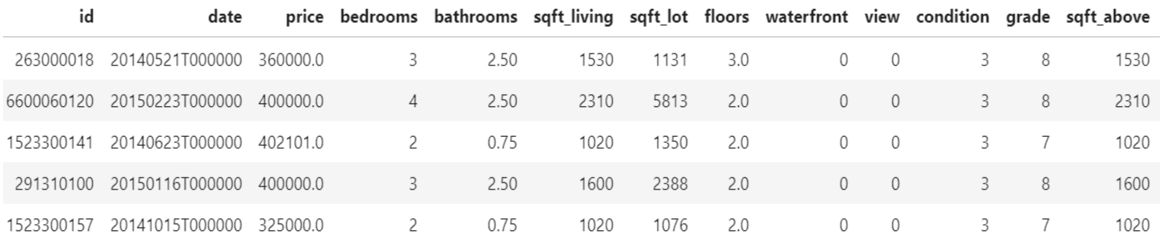
Attributes of the dataset:

The dataset has 20 features:



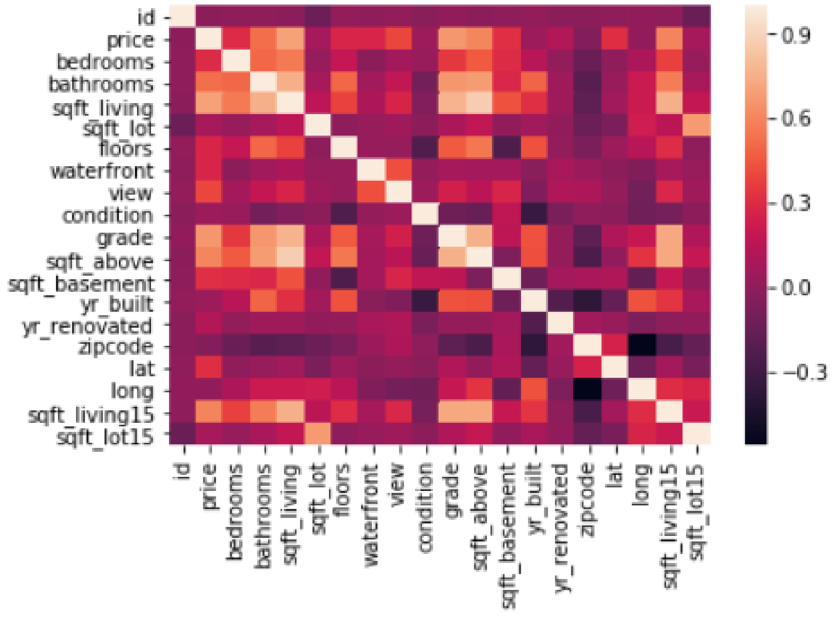
```
id              14203 non-null int64
date            14203 non-null object
price           14203 non-null float64
bedrooms        14203 non-null int64
bathrooms       14203 non-null float64
sqft_living     14203 non-null int64
sqft_lot        14203 non-null int64
floors          14203 non-null float64
waterfront      14203 non-null int64
view            14203 non-null int64
condition       14203 non-null int64
grade           14203 non-null int64
sqft_above      14203 non-null int64
sqft_basement   14203 non-null int64
yr_built        14203 non-null int64
yr_renovated    14203 non-null int64
zipcode         14203 non-null int64
lat             14203 non-null float64
long            14203 non-null float64
sqft_living15   14203 non-null int64
sqft_lot15      14203 non-null int64
```

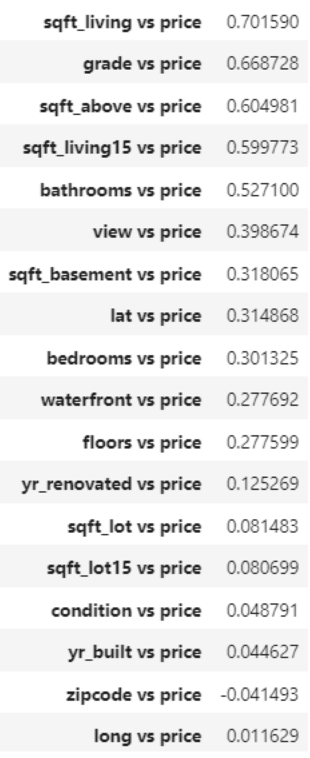
First few rows of the data:



id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above
263000018	20140521T000000	360000.0	3	2.50	1530	1131	3.0	0	0	3	8	1530
6600060120	20150223T000000	400000.0	4	2.50	2310	5813	2.0	0	0	3	8	2310
1523300141	20140623T000000	402101.0	2	0.75	1020	1350	2.0	0	0	3	7	1020
291310100	20150116T000000	400000.0	3	2.50	1600	2388	2.0	0	0	3	8	1600
1523300157	20141015T000000	325000.0	2	0.75	1020	1076	2.0	0	0	3	7	1020

Finding Correlation within the features by visualizing Correlation plot:





sqft_living vs price	0.701590
grade vs price	0.668728
sqft_above vs price	0.604981
sqft_living15 vs price	0.599773
bathrooms vs price	0.527100
view vs price	0.398674
sqft_basement vs price	0.318065
lat vs price	0.314868
bedrooms vs price	0.301325
waterfront vs price	0.277692
floors vs price	0.277599
yr_renovated vs price	0.125269
sqft_lot vs price	0.081483
sqft_lot15 vs price	0.080699
condition vs price	0.048791
yr_built vs price	0.044627
zipcode vs price	-0.041493
long vs price	0.011629

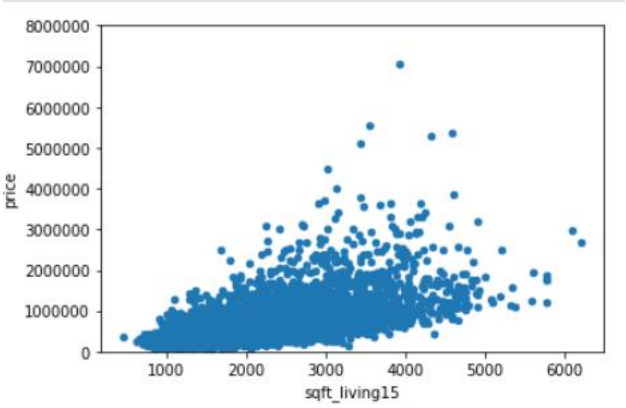
Feature Selection:

Because zipcode is negatively correlated with sales price, we can ignore it when predicting sales price.

Report:

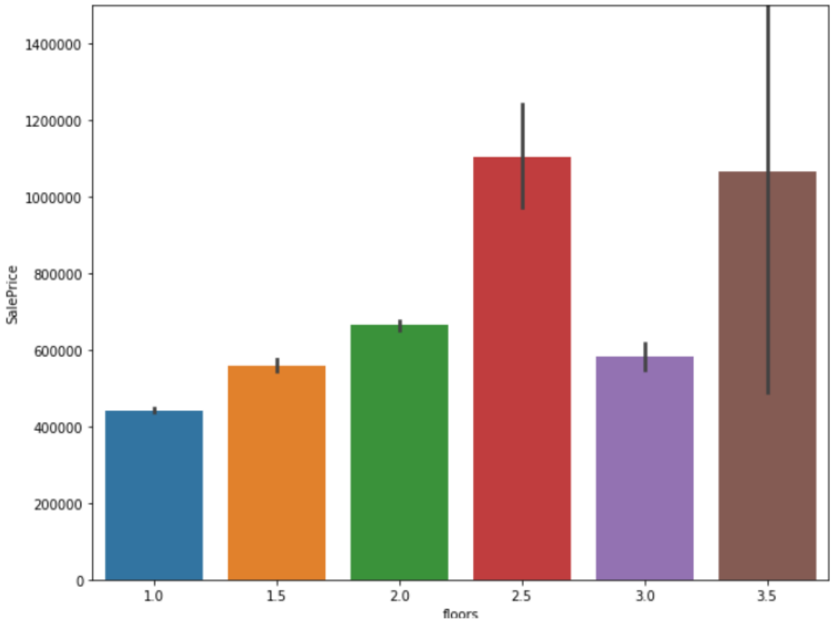
Visualization of Data:

Price vs sqftliving15 visualization-



We can infer from the plot that majority of the houses with living area between 1000 and 4000 are priced under $2000000.

Price VS Floors-



The above plot explains the relation between number of floors of the house and the price of the house.

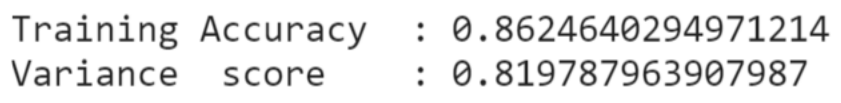
Splitting Data-

The predictors or variables used in the training and test datasets must be comparable. In terms of observations and variable values, they diverge. Minimizing error or identifying the right answers is implicitly achieved by fitting the model to the training dataset. The fitted model performs well when applied to the training dataset. The test dataset is then used to evaluate the model. If the model also makes accurate predictions on the test dataset, your analysis is good. Because the test dataset and training dataset are comparable but not identical or visible to the model, you can be more confident. It indicates that the model actually performs learning or prediction.

Here the dataset was split in the ratio of 25% for testing and 75% for training.

Results:

The model yielded accuracy of 86% with variance score of 81.9 which is close to 1. The Random Forest performed well according to my analysis.



```
Training Accuracy  : 0.8624640294971214
Variance  score    : 0.819787963907987
```

References:

Geeksforgeeks - [random-forest-regression]

Towards Data Science - [Bias vs Variance]

Dataset - [House Sales | Kaggle]

Professor Chen's Lecture Slides.

2. AdaBoost Classifier:

In 1996, Yoav Freund and Robert Schapire proposed the Ada-boost ensemble boosting classifier. It combines several classifiers to improve classifier accuracy. AdaBoost is an iterative ensemble method. The AdaBoost classifier combines a number of ineffective classifiers to produce a powerful classifier with high accuracy. Adaboost's fundamental idea is to train the data sample and set the classifier weights in each iteration in such a way that accurate predictions of unusual observations are made. As the base classifier, any machine learning algorithm that accepts weights from the training set can be used.

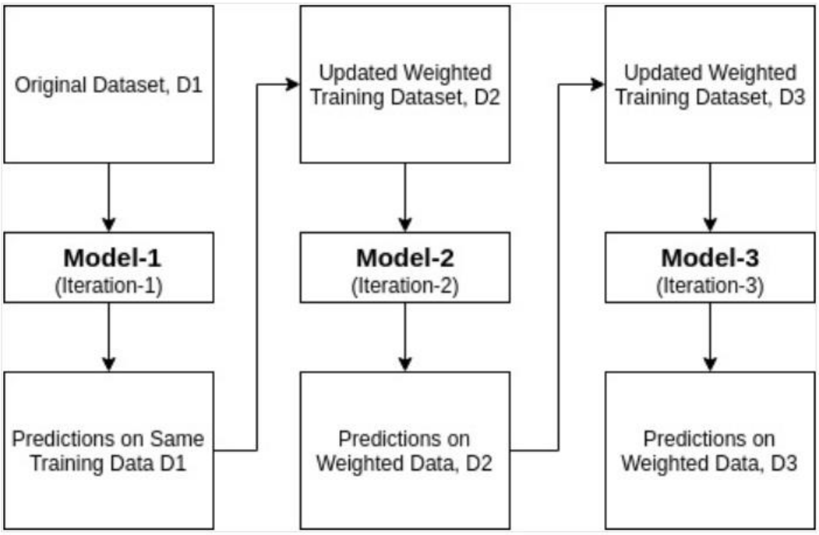
Adaboost must obey two rules:

- Several different weighed training examples should be used to interactively train the classifier.
- It works hard to reduce training error in order to provide the best possible fit for these examples in each iteration.

Algorithm-

It functions in the following way:

- Adaboost starts by randomly choosing a training subset.
- By choosing the training set based on the efficacy of the prior training, it iteratively trains the AdaBoost machine learning model.
- It increases the weight of incorrectly classified observations, increasing their likelihood of classification in the following round.
- Additionally, based on the trained classifier's accuracy, weight is given to it in each iteration. There will be more emphasis placed on the more accurate classifier.
- Repeat this procedure until all training data fits accurately or until the specified maximum number of estimators is reached, whichever comes first.
- Vote among all of the learning algorithms you produced in order to categorize.


Original Dataset, D1
Updated Weighted Training Dataset, D2
Updated Weighted Training Dataset, D3
Model-1 (Iteration-1)
Model-2 (Iteration-2)
Model-3 (Iteration-3)
Predictions on Same Training Data D1
Predictions on Weighted Data, D2
Predictions on Weighted Data, D3

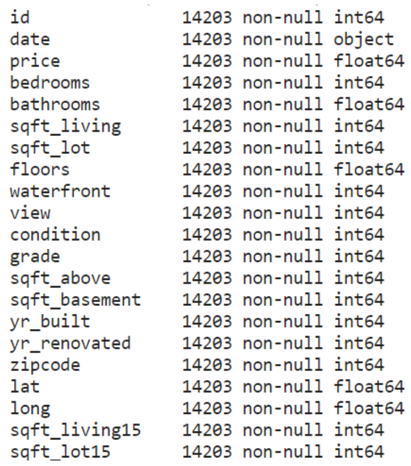
Dataset:

The dataset used for random forest is used for AdaBoost as well.

We must project the sales prices of homes in the county of Seattle using this dataset. Properties bought between 2014 and 2015 are included.

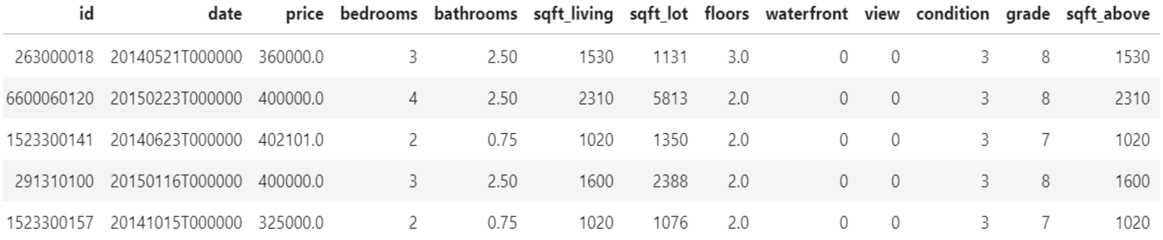
Attributes of the dataset:

The dataset has 20 features:



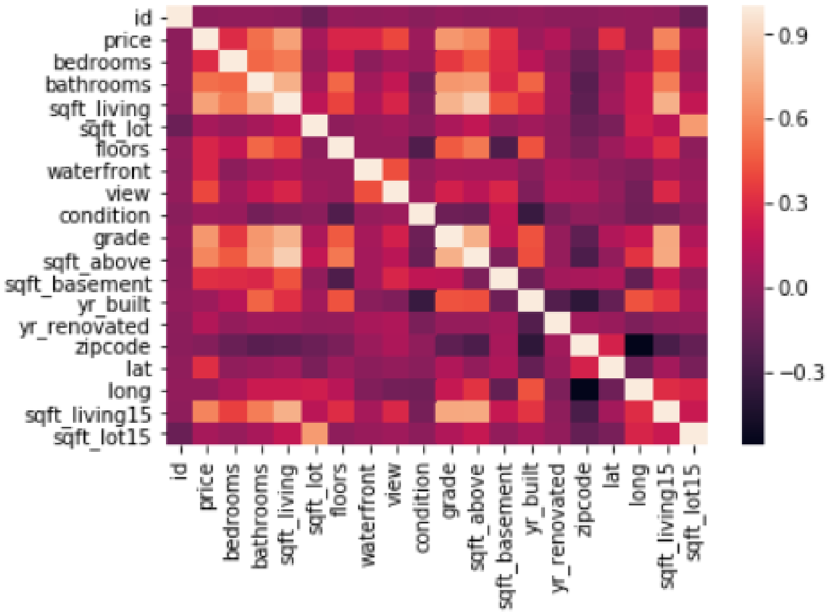
```
id               14203 non-null int64
date             14203 non-null object
price            14203 non-null float64
bedrooms         14203 non-null int64
bathrooms        14203 non-null float64
sqft_living      14203 non-null int64
sqft_lot         14203 non-null int64
floors           14203 non-null float64
waterfront       14203 non-null int64
view             14203 non-null int64
condition        14203 non-null int64
grade            14203 non-null int64
sqft_above       14203 non-null int64
sqft_basement    14203 non-null int64
yr_built         14203 non-null int64
yr_renovated     14203 non-null int64
zipcode          14203 non-null int64
lat              14203 non-null float64
long             14203 non-null float64
sqft_living15    14203 non-null int64
sqft_lot15       14203 non-null int64
```

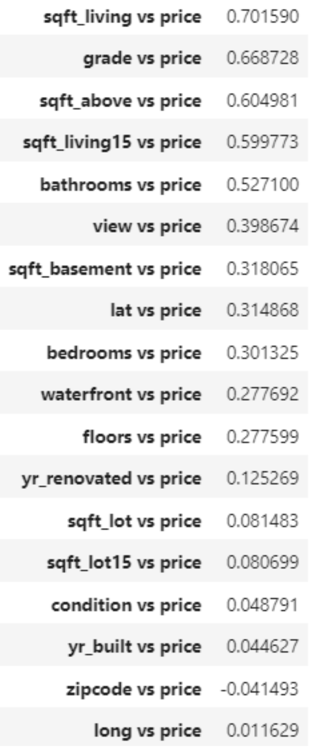
First few rows of the data:



id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above
263000018	20140521T000000	360000.0	3	2.50	1530	1131	3.0	0	0	3	8	1530
6600060120	20150223T000000	400000.0	4	2.50	2310	5813	2.0	0	0	3	8	2310
1523300141	20140623T000000	402101.0	2	0.75	1020	1350	2.0	0	0	3	7	1020
291310100	20150116T000000	400000.0	3	2.50	1600	2388	2.0	0	0	3	8	1600
1523300157	20141015T000000	325000.0	2	0.75	1020	1076	2.0	0	0	3	7	1020

Finding Correlation within the features by visualizing Correlation plot:





sqft_living vs price	0.701590
grade vs price	0.668728
sqft_above vs price	0.604981
sqft_living15 vs price	0.599773
bathrooms vs price	0.527100
view vs price	0.398674
sqft_basement vs price	0.318065
lat vs price	0.314868
bedrooms vs price	0.301325
waterfront vs price	0.277692
floors vs price	0.277599
yr_renovated vs price	0.125269
sqft_lot vs price	0.081483
sqft_lot15 vs price	0.080699
condition vs price	0.048791
yr_built vs price	0.044627
zipcode vs price	-0.041493
long vs price	0.011629

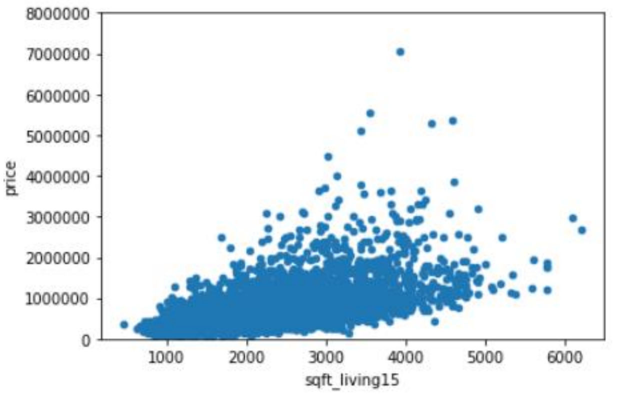
Feature Selection:

Because zipcode is negatively correlated with sales price, we can ignore it when predicting sales price.

Report:

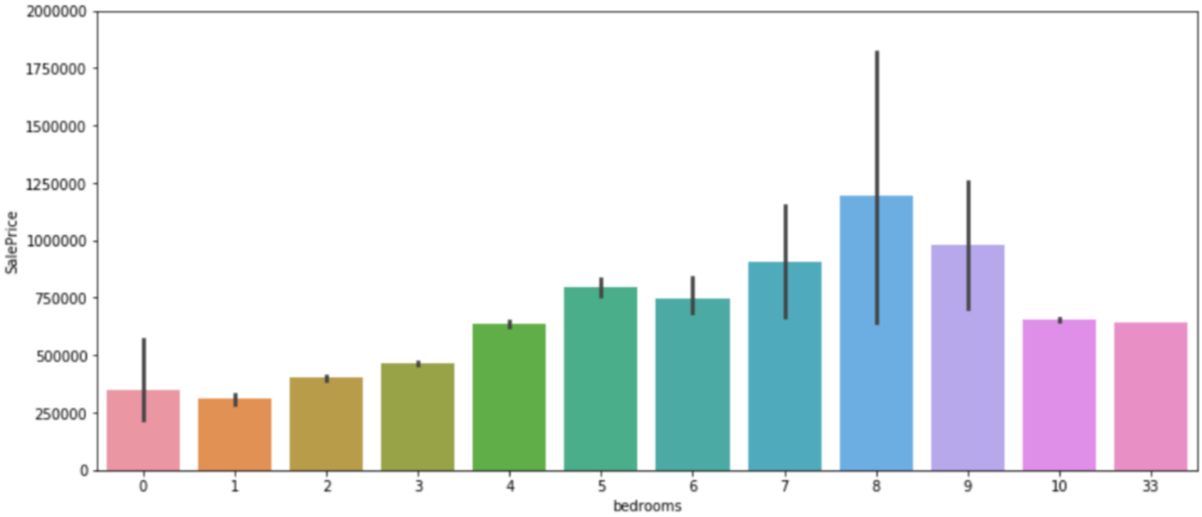
Visualization of Data:

Price vs sqftliving15 visualization-



We can infer from the plot that majority of the houses with living area between 1000 and 4000 are priced under $2000000.

Price VS Bedrooms:



The above plot explains the relation between number of bedrooms and the price of the property

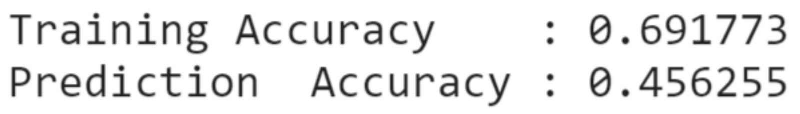
Splitting Data-

The predictors or variables used in the training and test datasets must be comparable. In terms of observations and variable values, they diverge. Minimizing error or identifying the right answers is implicitly achieved by fitting the model to the training dataset. The fitted model performs well when applied to the training dataset. The test dataset is then used to evaluate the model. If the model also makes accurate predictions on the test dataset, your analysis is good. Because the test dataset and training dataset are comparable but not identical or visible to the model, you can be more confident. It indicates that the model actually performs learning or prediction.

Here the dataset was split in the ratio of 30% for testing and 70% for training.

Results:

The model yielded accuracy of 69% with prediction score of 45 which is not close to 1. The Random Forest performed well when compared to AdaBoost according to my analysis



```
Training Accuracy     : 0.691773
Prediction  Accuracy : 0.456255
```

References:

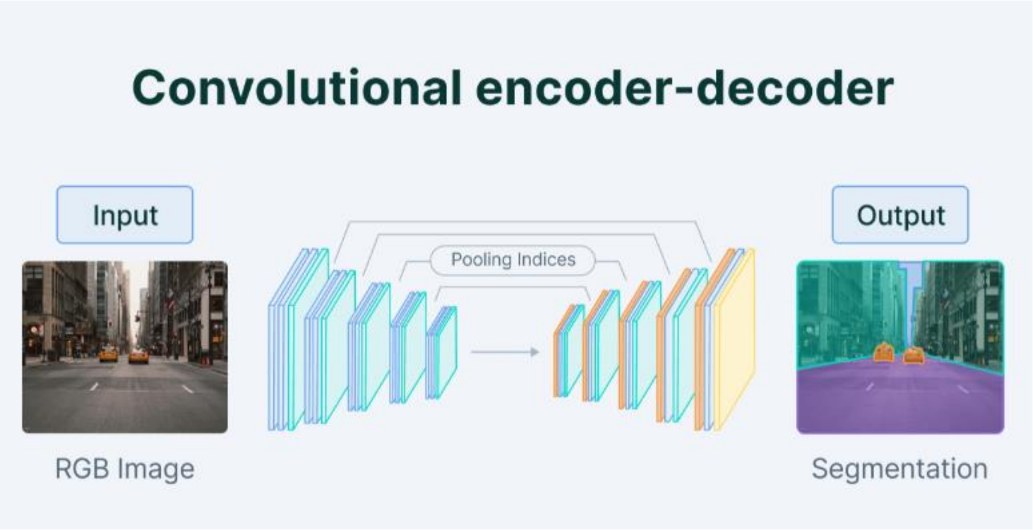
Data Camp - AdaBoost

Dataset   -   House Sales | Kaggle

Professor Chen’s Lecture Slides.

3. Autoencoder

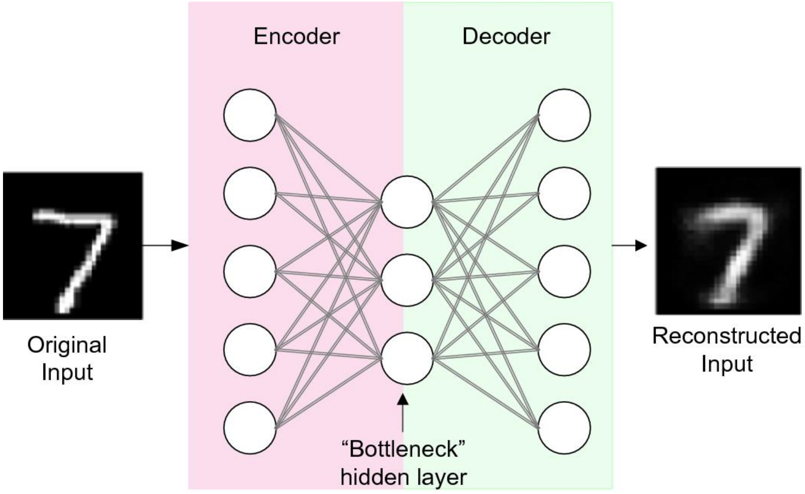
To unsupervised learn data encodings, an artificial neural network called an autoencoder is used.
An autoencoder's goal is to train the network to capture the most important elements of the input image so that it can learn a lower-dimensional representation (encoding) for a higher-dimensional data, typically for dimensionality reduction.



Algorithm:

Autoencoder has 3 components-

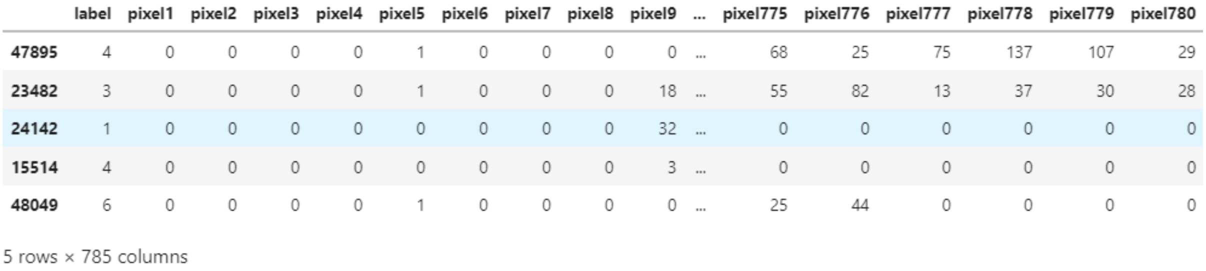
- Encoder: A component that shrinks the train-validate-test set input data into an encoded representation that is typically many orders of magnitude smaller than the input data.
- Bottleneck: The most crucial element of the network because it contains the compressed knowledge representations.
- Decoder: A component that helps the network "decompress" knowledge representations and reconstruct encoded data. The output is then measured against a standard value.



Dataset:

The training and testing data files. csv. Grayscale images of the digits 0 to 9 are included in CSV files.

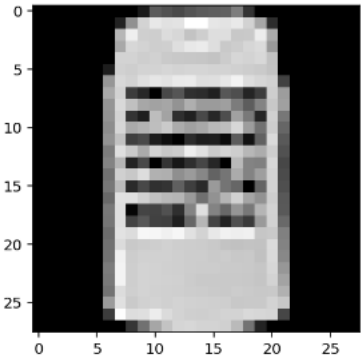
The combined height and width of each image are 28 pixels, making a total of 784 pixels. The lightness or darkness of each pixel is indicated by its single pixel value, with higher numbers indicating darker pixels. This pixel value is an integer between 0 and 255, inclusive.



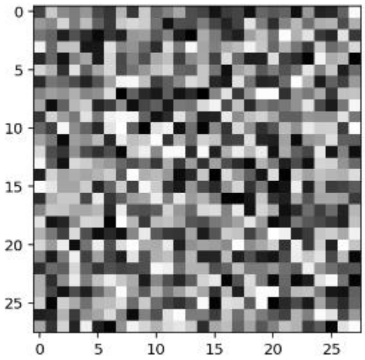
	label	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780
47895	4	0	0	0	0	1	0	0	0	0	...	68	25	75	137	107	29
23482	3	0	0	0	0	1	0	0	0	18	...	55	82	13	37	30	28
24142	1	0	0	0	0	0	0	0	0	32	...	0	0	0	0	0	0
15514	4	0	0	0	0	0	0	0	0	3	...	0	0	0	0	0	0
48049	6	0	0	0	0	1	0	0	0	0	...	25	44	0	0	0	0

5 rows × 785 columns

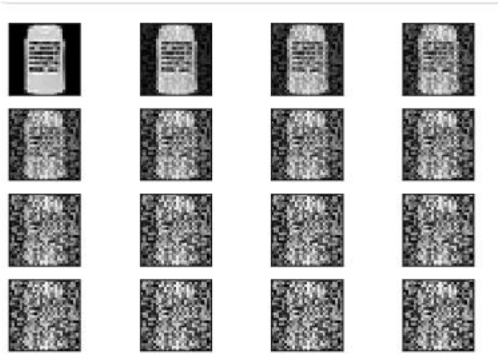
From the dataset, I chose the following image to be encoded, lets consider it as window image:



Unless we are extremely lucky, no matter how many times we try to sample at random, all we will ever get is static and nothing that even remotely resembles a real digit. This is compelling empirical evidence that meaningful images—in this case, images of numbers—are clustered in smaller dimensional subsets in the original 784-dimensional pixel space. This is referred to by the manifold hypothesis. The promise is that developing machine learning systems will be easier if we understand the structure of the manifold better.

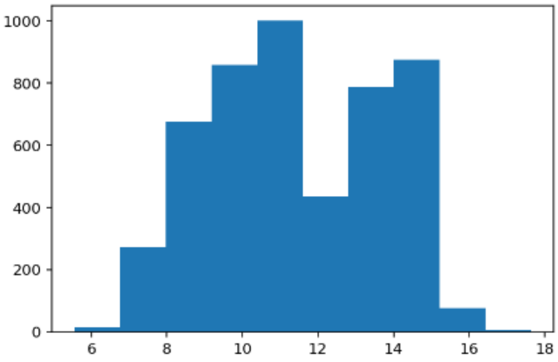


Moving away from an image in the 784-dimensional image space in a random direction:



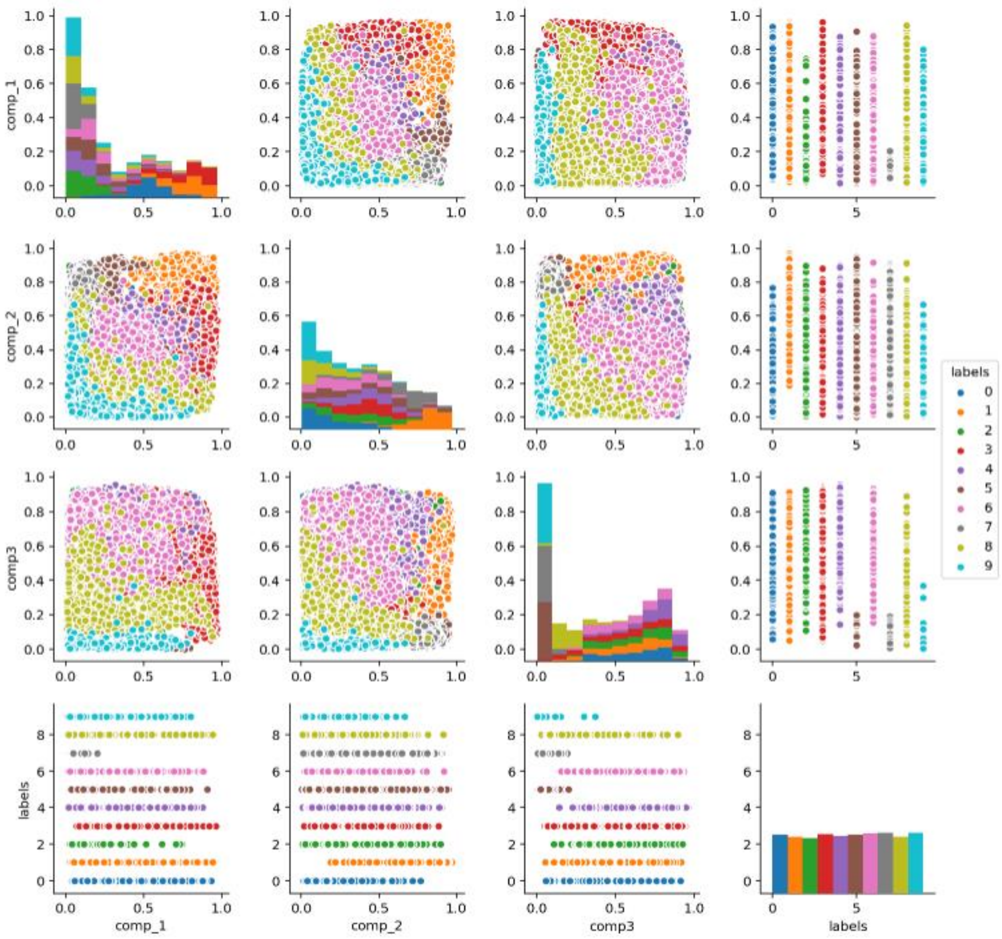
So we can see that as we move away from the window image, the images become less and less distinct. We can still see the shape of an eight at first, but before we know it, we've returned to static.

Utilizing some metric to determine which images are closest to the "eight" image will help you better understand the structure of the image space. I'll apply the sklearn knn wrap to the flattened images using l 2 distance as the metric.



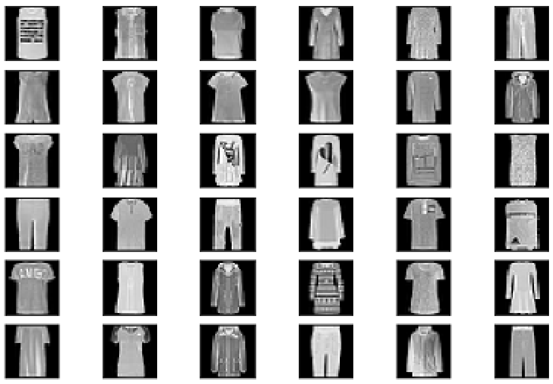
The separations from the "window" image in the first 5000 images are roughly normally distributed; in fact, they behave much better than I expected. Because we have different classes, I expected multiple modes and a higher variance.

Visualization using Seaborn:



RESULTS:

The following are all the images which are segmented according to our window image-



References:

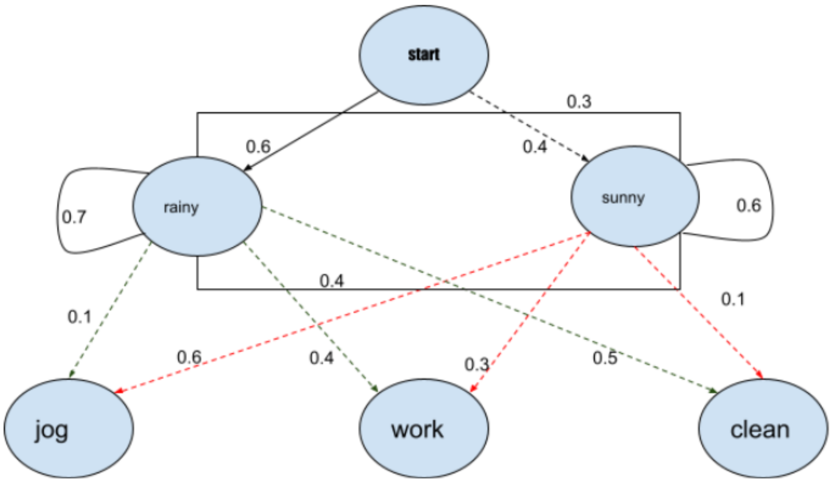
DATASET - Digit recognizer | Kaggle

Autoencoder - V7 Labs

Professor Chens Lecture videos and notes.

4. Hidden Markov Model

Statistical models known as hidden Markov models (HMMs) have been around for a while. They have been applied in a number of disciplines, such as data science, computer science, and medicine. The Hidden Markov model serves as the foundation for many contemporary data science algorithms. In data science, it has been used to effectively use observations in order to make precise predictions or decisions. This article will discuss hidden Markov models using relevant examples from everyday life and key hidden Markov model ideas.



In the early 1900s, Andrey Markov developed the first Markov models, hence the name. A Markov model is a type of probabilistic model that is used to forecast a system's future state from its current state. In other words, Markov models are used to forecast the future state based on the current hidden or observed states. The Markov model is a finite-state machine with the possibility of changing to any other state after one step. They can be used to simulate real-world problems with hidden and observable states. Markov models are classified as hidden or observable based on the type of information that can be used to make predictions or decisions.

Dataset:

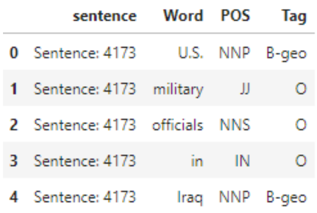
Context

This is a clean dataset and called Named Entity recognition task of Natural Language Processing

Content

The dataset has 1M x 4 dimensions is grouped by #Sentence and contains columns = ['# Sentence', 'Word', 'POS', 'Tag'].

Sample rows from data:

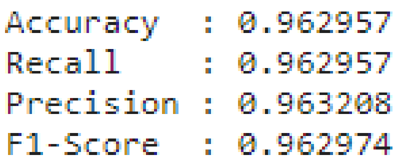


	sentence	Word	POS	Tag
0	Sentence: 4173	U.S.	NNP	B-geo
1	Sentence: 4173	military	JJ	O
2	Sentence: 4173	officials	NNS	O
3	Sentence: 4173	in	IN	O
4	Sentence: 4173	Iraq	NNP	B-geo

Results:

To recognize the name entity from the sentences, the NER dataset is used.

The model performed really well with accuracy and precision of 96%.



```
Accuracy  : 0.962957
Recall    : 0.962957
Precision : 0.963208
F1-Score  : 0.962974
```

References:

Dataset - NER Dataset | Kaggle

HMM - HMM | Vitalflux

Professor Chens Lecture videos and notes.