**8.1.3- Linear regression Mini project:**

Linear regression equation Y = Beta (B0) +B1\* X + €

X is known as independent variable, **explanatory variable, features, predictors**

Y is dependent variable, response variable

B0 is intercept and represents average of Y when all independent variables X are set to 0

B1 is the slope of the line, and represents average effect of one unit increase of X on YE

Residual: In regression analysis, the difference between actual value of dependent variable and predicted value is called residual € = y – yhat

Residual plot: is a graph that shows the residuals on the vertical axis and independent variable on the horizontal axis

Assumptions made in Linear regressions: Y = Beta (B0) +B1\* X + €

1. Epsilon € : it is unobservable random variable that adds noise to the linear relation, € is assumed to be normally distributed with mean of 0
2. The residuals € are also assumed to be independently and identically distributed, i.e the residuals from one prediction do not have effect on another prediction

Our main focus in linear regression is to estimate B0 and B1, we will mostly use LEAST SQUARED method to estimate them, once we estimate B0 and B1, then we can calculate Y (estimated Y is called Y CAP), based on new values of X

LEAST SQUARED METHOD:

*ℒ*=∑*i*=1*Nϵ*2*i*

=∑*i*=1*N*(*yi*−*y*̂*i*)2

=∑*i*=1*N*(*yi*−(*β*0+*β*1*xi*))2

We want to find B0 and B1, that minimizes the squared error, so first we will do partial derivative w.r.t B0

If you do that you will get B0 = Y MEAN + B1\* X MEAN

Similarly, if take partial derivative w.r.t B1 and substituting above value, you will get following matrix notation

*β*̂=(*XTX*)−1*XTY*

**Linear regression using STATSMODEL:**

import statsmodels.api as sm

from statsmodels.formula.api import ols

m=ols(‘PRICE’ ~ ’RM’, bos).fit()

print(m.summary())

for statsmodels (ols or logit) calls you have a pandas data frame with column names that you will add to the formula.

You can force statsmodels to treat variables as categorical with the `C()` function, call numpy functions to transform data such as `np.log` for extremely-skewed data, or fit a model without an intercept by including `- 1` in the formula. For a quick run-down of further uses see the `statsmodels` [help page](http://statsmodels.sourceforge.net/devel/example\_formulas.html).

**Linear regression using SCIKIT LEARN:**

from sklearn.linear\_model import LinearRegression

X = bos.drop('PRICE', axis = 1)

# This creates a LinearRegression object

lm = LinearRegression()

lm

**Interpreting residual plots, to improve regression** - <http://docs.statwing.com/interpreting-residual-plots-to-improve-your-regression/>

**Interpreting R-Squared values** - <https://statisticsbyjim.com/regression/interpret-r-squared-regression/>

**Linear regression by Stat quest:** (StatQuest: Linear Models Pt.1 - Linear Regression)

1. Use least-squares to fit a line to the data
2. Calculate R square
3. Calculate p-value for R-square

Calculating R2 is first step in determining how good a guess(model) is.

R2 = (Var(mean)-Var(fit))/Var(mean)

Var(mean) = Variance around the mean of Y

Var(fit) = Variance around the predicted line

For example if we take mouse size (y-axis) and weight (x-axis) example (i.e. we are estimating size based on weight) and we calculate R2 to 0.6, i.e. 60%, it specifies that we can explain 60% of variance in mouse size by Weight.

**8.1.4 – Logistic regression behind the scenes**

**Github.com/moody-marlin/pydata\_logistics**

<http://www.stat.yale.edu/Courses/1997-98/101/ranvar.htm> -> Random variables, discrete random variable, probability mass function, continuous random variable, cumulative distribution function,

My understandings from above site:

Random variable: is an outcome of a random phenomenon, i.e. any of the possible outcomes and there are two types

Discrete Random variable: i.e. if the outcome is countable, i.e. no of children in a family etc

Continuous Random variable: examples height, weight, amount of sugar in orange, the value can be anything.

Probability mass function: is list of probabilities of all possible outcomes of a discrete random variable.

Joint Probability mass function: JPMF of discrete random variables X and Y is P(X=x, Y=y) i.e P(X=x) and P(Y=y)

Cumulative Distributive Function: it is a function giving the probability of random variable X which is less than x. this is provided for both Discrete and Continuous Random variables.

Maximum likelihood: The goal of maximum likelihood is to find an optimum way to fit a distribution to the data.

**Classical stats vs Bayesian statistics:**

Classical stats uses techniques such as Ordinary Least squares and Maximum Likelihood - this is the conventional type of statistics that you see in most text books.

Bayesian statistics looks quite different and this is because it is all about modifying conditional probability – it uses prior distributions for unknown quantities which it then updates to posterior distribution using the laws of probability.

<https://egertonconsulting.com/a-comparison-of-classical-and-bayesian-statistics/?doing_wp_cron=1582417278.8493371009826660156250>

Asymptotic: is a line that approaches a curve but never touches it.

**Limiting distribution or Asymptotic Distribution**: ??? did not understand

**Inference vs prediction:**

Inference: Use the model to learn about data generation process

Prediction: Use the model to predict outcome for new data points.

**8.1.5 Logistic Regression**

I think logistic regression is used for classification problems

There are different ways of making classifications one such way is **Maximum Margin classifier**: in this approach we construct a decision boundary that is as far as possible away from both classes of points.

The fact that a line can be drawn to separate the two classes makes the problem **linearly separable**.

Support Vector Machines are examples of Maximum Margin Classifier

SKLEARN.LINEAR\_MODEL.LOGISTICREGRESSION:

Basically, Logistic regression is used only for binary classifiers, but you can use clever extensions to logistic regression, to make it useful for Multiclass case, I think there are two ways

1. One vs Rest(OVR) :- <https://chrisalbon.com/machine_learning/logistic_regression/one-vs-rest_logistic_regression/>. In OVR a separate model is trained for each output class, the model can predict if an observation belongs to that class or not, so we must build a separate model for each output class.
2. Cross-entropy loss (multinomial) : <https://machinelearningmastery.com/cross-entropy-for-machine-learning/>

Entropy: - Information too can be measured and compared using a measurement called entropy. Think of it as an information scale. We intuitively know that a single page from some unknown book has less information than the entire book. We can describe exactly how much using a unit called the bit

Cross Entropy: - it is a measure of the difference between two probability distributions for a given random variable or sets of events.

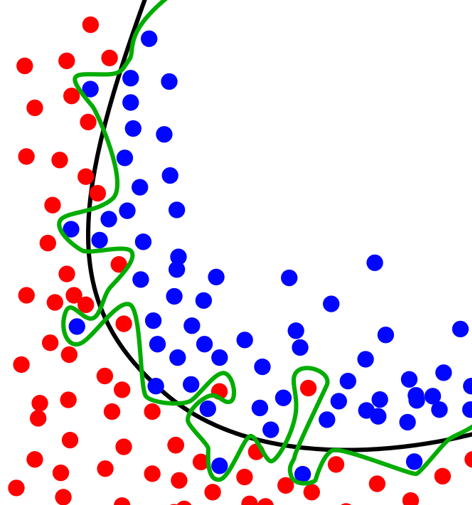
Information Theory: - information theory studies the quantification, storage and computation of information.

Loss Function: - At its core loss function is simple: it’s a method of evaluating how well your algorithm models your dataset. If your predictions are totally off, your loss function will output high number, if they are pretty good, it will output lower number.

<https://algorithmia.com/blog/introduction-to-loss-functions>

**Sampling error**: In Statistics sampling errors are incurred when the statistical characteristics of a population is estimated from a sample of that population, because of this there is a chance that the characteristics of the sample, like mean, quartiles, generally differ from the characteristics of the entire population. i.e. average height of thousand people is not always same as average height of one million people. (Wiki)

**Overfitting**: is estimating the model too closely, it fits very well for the given sample data, but may cause greater error than regularized model. Similarly, **Underfitting**. These two can also be called as Overtraining and Undertraining. (Wiki)



**Generalization error**: in supervised learning applications in machine learning and statistical learning theory, generalization theory is a measure of how accurately an algorithm can predict outcome values for previously unseen data. Generalization error can be minimized by avoiding overfitting in the learning algorithm. The performance of a machine learning algorithm is measured by plots of the generalization error values through the learning process, which are called learning curves.

Training Error:

**Regularization of Logistic regression**: Regularization can be used to avoid overfitting. Regularization is any modification, we make to a learning algorithm, that is intended to reduce its generalization error, but not its training error

**BIAS**:- In [statistics](https://en.wikipedia.org/wiki/Statistics), the **bias** (or **bias function**) of an [estimator](https://en.wikipedia.org/wiki/Estimator) is the difference between this estimator's [expected value](https://en.wikipedia.org/wiki/Expected_value) and the true value of the parameter being estimated. An estimator or decision rule with zero bias is called **unbiased**.

**ODDS**: Is defined as probability of success divided by probability of failure. i.e. p/(1-p)

For example assume that probability of rain tomorrow is 80%, then ODDS of raining tomorrow is 0.8/0.2, i.e. 4

**LOG OF ODDS**:- Log of odds is logarithm of odds, i.e. in above example it is Ln(4)

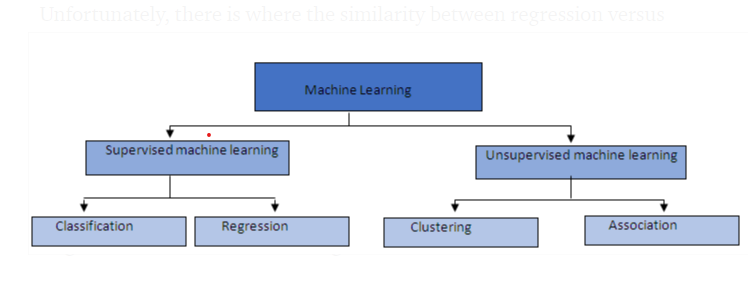
**LOGIT**:- LOGIT(P) = LOG(P-(1-P)) (LOG OF ODDS). I think both LOG OF ODDS and LOGIT are same.

**8.2.1 – Decision Trees for Classification and Regression:** The video provided is not an introduction, they just talk something random. Below site will give you slight idea

<https://towardsdatascience.com/decision-tree-classification-de64fc4d5aac>

Difference between Classification and Regression: Output variable in regression in numerical (or continuous) while for Classification it is Categorical or Discrete

<https://medium.com/quick-code/regression-versus-classification-machine-learning-whats-the-difference-345c56dd15f7#:~:text=>



**8.2.4 Gradient Boosting and XG Boost:**

SVM – Support Vector Machines (Hinge Loss)

RBF – Radial Basis Function Kernel (Kernelized learning algorithms)

bias and variance are better explained in below link stat quest videos as well <https://medium.com/datadriveninvestor/bias-and-variance-in-machine-learning-51fdd38d1f86>

**Bagging (bootstrap aggregating)**:from data set (D) of size N, we choose N prime sampels (generally 60% of data) and send it to model (in general Decision trees, but we can use any modeling technique), we repeat this every time with different randomly selected data and then take average (for regression) and voting for (selection).

I think Bagging can run parallelly

**Boosting**: boosting is also similar to Bagging, it tries to combine weak predictors and produce strong predictors, first we select random data and run it through the model and test it again on the same data, and the points for whcih the model did not work properly it will assign more weight, and then we select data for next round (the probability to get picked is more for these weighted points) and then again run it through the model.

The major difference is we are not selecting data randomly, we are trying to pick the data points for whcih the model did not work properly and model the next one. Also Boosting is sequential process, so we can not run them in parallel

There are various kinds of boosting,

1. Ada Boosting (Adaptive Boosting)
2. Gradient Boosting
3. XG Boosting
4. Cat Boosting

**STUMP**: a tree with one node and two leaves. Stumps are very weak learners, as they only use one feature

Ordering of nodes for splitting criteria i.e. which feature we need to select first for creating a stump and then what, i.e. the priority of features we need to select, can be identified in two ways 1) entropy 2) Gini Index

1. **Entropy Function**: -(P+ \* log base 2 P+) –(P- \*log base 2 P-) , here P+ is probability of Positive sample and P- is probability of negative sample. if the values are homogeneous the value will be 0, if not it will be 1.Entropy will tell you how homogeneous the samples are. i.e. after splitting the data, it will if the split is as expected or not.

**Information Gain**: S (starting entropy)-7/14\*0.985 - 7/14\*0.592 = 0.151 , the one with maximum gain is used for splitting criteria

example: 9 yes and 5 no: starting entropy is -9/14 \* log (9/14) -5/14 \* log(5/14) = 0.94

(3+, 4-) E=0.985 and (6+, 1-) E=0.592

1. **GINI INDEX:** if a data set contains n classes, gini index is 1 - sum (1 to n) of squares of probabilities, if a data set is split on A into two subsets D1 and D2, then gini(D) is defined as no(D1)/no(D) \* gini(D1) + no(D2)/NO(D) \* gini(D2)

**Adaptive Boosting:** AdaBoost follows following three points

1. AdaBoost combines lots of weak learners to make classifications, the weak learners are always stumps
2. Sum stumps get more say in the final classifications than others, i.e. the stumps are weighted.
3. Each stump is created by taking previous stumps mistakes into account, so the order of stumps is important, as error in first stump influences how the second stump is made, so we use either entropy function of Gini index to decide the order.

Detailed explanation is present in below videos by Stats quest

<https://www.youtube.com/watch?v=LsK-xG1cLYA>

To identify how much a stump has in the final say, we do the following this is also explained in above video, but I am writing it down any way

**AMOUNT OF SAY:** take an example of a stump with (3+, 0-) and (4+, 1-), firs we will calculate total error

Total Error: -ve \* weight. Initially the weight will be 1/no of records, so in above example we have 8 records, so weight of each record is 1/8. So total error is 1\*1/8

Amount of say = 0.5 \* log ((1-total error)/total\_error) = 0.97

The amount of say varies from -3 to +3, with greater the value greater the say.

Modifying Sample Weight after each stump:

1. Increasing the weight for incorrectly classified sample (record)

New Sample Weight = sample weight \* (e power (amount of say))

= 1/8 (e power 0.97) = 0.33 (>0.125)

1. Decrease the weight for correctly classified samples (records)

New Sample Weight = sample weight \* (e power (-1 \* amount of say))

= 1/8 (e power -0.97) = 0.05

1. Normalize the weights: If we add 0.33 + 7\*0.05 = 0.68, this is not equal to 1. So we will normalize the weights. So divide each sample weight by 0.68, we will get normalized sample weights

If it’s a classification, then we just add the Amount of say of all the stumps that say yes and similarly No, we choose the maximum value.

**GRADIENT BOOSTING: (Stat Quest)**

Gradient boosting can be used for both Regression and Classification, it is very similar to AdaBoost, in this we will consider an example of (Height, favorite color, Gender) and (Weight).

1. It starts by making single leaf : in above example it will be average of weights.
2. It builds a tree, similar to AdaBoost, it is build based on errors in previous step, and unlike AdaBoost, this is not stump, but actual full tree (though you can limit no of leaves in general between 8 and 32). Similar to AdaBoost Gradient Boost weighs the trees, but all the trees are given same weight and this is called Learning Rate, whose value is between 0 and 1.
3. In this example the error in previous tree, is the difference between observed weight and predicted weight.

**Regularization: Ridge Regression (Stat quest)**

The main idea behind Ridge Regression is to find a new line that doesn’t fit the training data too well, that is to reduce overfit, this is done by adding small amount of Bias and in return there will be significant drop in variance.

Least squares Minimizes sum of squares of Residuals

Ridge regression Minimizes the sum of squares of residuals + Lambda \* slope Square

How Ridge Regression works:

1. Find out Least Squares Line and calculate sum of squares of Residuals + Lambda \* Slope Square
2. Find out Ridge Regression Line and calculate SSR + Lambda \* Slope Square
3. Smaller the value, better the line
4. Lambda can be any value from 0 to + infinity, Lambda \* Slope Square is called **Ridge Regression Penalty**
5. How to decide Lambda Value: we just try bunch of values for Lambda and use Cross Validation, typically 10 fold Cross Validation, to determine which one results in the lowest variance

Once

**XG BOOST:** (stat quest)

It is extreme Gradient Boosting and it means it is a big Machine Learning algorithm with lots of parts, please find them below, each part is easy to understand

1. Gradient Boost
2. Regularization
3. A unique Regression Tree
4. Approximate Greedy Algorithm
5. Weighted Quantile Sketch
6. Sparsity-Aware Split Finding
7. Parallel Learning
8. Cache-Aware Access
9. Blocks for Out-of-core computation

XGBoost was designed to be used with large complicated data sets.

It uses unique Regression Tree known as XGBoost Tree.

I believe this is a huge concept and no need to know right away, we can learn about these once I land in a job.

**IMPORTANT CODING**

**Code to implement Scatter plot:**

import matplotlib.pyplot as plt

plt.scatter(q.fittedvalues q.resid, c=’Red’, s=10, alpha=0.2)

plt.xlabel("fitted values")

plt.ylabel("residuals")

plt.title("Residual Plot with three varialbes")

plt is alias for matplotlib.pyplot

q.fittedvalues (x-axis) and q.resid(y-axis) are columns of a dataframe, so they can be series.

c=’Red’ 🡪 set output scatter color to Red

s=10 🡪 size of each scatter , minimum is 0, will not display anything

alpha=0.2 🡪 transparency, minimum is 0 and maximum is 1.

**To split training and test data set:**

Xlr, Xtestlr, ylr, ytestlr = train\_test\_split(dflog[['Height','Weight']].values, (dflog.Gender == "Male").values,random\_state=5)

Splits arrays or matrices into random train and test subsets

Parameters:

1. Arrays: sequence of indexables with same length
2. Test\_size: default is 0.25
3. Random\_state: if int, then this is considered as seed for random number generator, what it means is if you use same integer number (Say 4), for same input it will always generate same split, how many number of times you run it.
4. I think we can also provide filter condition like dflog.Gender == “Male”

**Things to do**

Statistics:

1. Complete AP Statistics in Khan Academy, this might improve your understanding on statisicts

<https://www.khanacademy.org/math/ap-statistics>

**Mentor Call:**

**23 Feb 2020**

Questions:

1. What is p-value

**VOCABULARY**

1. Stochastic: randomly determined, used interchangeably with random process.
2. Ensembel :- A group of musicians, actors, or dancers who perform together.

A group of items viewed as a whole rather than individually.

1. Heuristic:- enabling a person to discover or learn something for themselves
2. Empirical:- verifiable by observation or experience rather than theory or pure logic.