**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

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/>

**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, continuous random variable, cumulative distribution function

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 have to 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>

**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