**DATA ANALYSIS WITH PYTHON**

**Introduction to Statistics**

**Mean, Median and Mode**

When it comes to Statistics, Mean, Median and Mode are the basic measures. Lets learn a little more about what they are, how are they calculated and when to use them and the syntax of calculating them using Python.

The **mean** is the average value in a collection of numbers. In statistics, it is a measure of central tendency of a probability distribution along median and mode. It is also referred to as an expected value. Mean can be calculated using the formula in the image.

When should you not use Mean?

The mean is usually the best measure of central tendency to use when your data distribution is continuous and symmetrical, such as when your data is normally distributed. However, it all depends on what you are trying to show from your data.

A diagram of a mathematical equation

Description automatically generated

**Median** is a statistical measure that determines the middle value of a dataset listed in ascending order (i.e., from smallest to largest value). Median can be calculated using the formula in the image.

When should you be using or not using Median?

The median is the most informative measure of central tendency for skewed distributions or distributions with outliers. For example, the median is often used as a measure of central tendency for income distributions, which are generally highly skewed.

What Is a Skewed Distribution?

 A distribution is said to be skewed when the data points cluster more toward one side of the scale than the other, creating a curve that is not symmetrical. In other words, the right and the left side of the distribution are shaped differently from each other.

A math equations and graphs

Description automatically generated with medium confidence

The **mode** is the most frequent score in our data set. On a histogram it represents the highest bar in a bar chart or histogram. You can, therefore, sometimes consider the mode as being the most popular option. Mode can be calculated using the formula in the image.

A mathematical equation with graphs and numbers

Description automatically generated with medium confidence

**Standard Deviation and Variance**

Standard deviation is a measure of dispersement in statistics. “Dispersement” tells you how much your data is spread out. Specifically, it shows you how much your data is spread out around the mean or average. Square root of the variance is the standard deviation. So what is Variance?

Variance measures how far a data set is spread out. It is mathematically defined as the average of the squared differences from the mean. The variance for a sample is calculated by:

1. Finding the mean(the average).
2. Subtracting the mean from each number in the data set and then squaring the result. The results are squared to make the negatives positive. Otherwise negative numbers would cancel out the positives in the next step. It’s the distance from the mean that’s important, not positive or negative numbers.
3. Averaging the squared differences.
4. Dividing the value by sample size - 1

**Z-Score**: A z-score (also called a standard score) gives you an idea of how far from the mean a data point is. But more technically it’s a measure of how many standard deviations below or above the mean a score is.

**Correlation**

Explains the relationship between two variables. Correlation coefficients are used to measure how strong the relationship is. Correlation coefficient formulas are used to find how strong a relationship is between data. The formulas return a value between -1 and 1, where:

* 1 indicates a strong positive relationship.
* -1 indicates a strong negative relationship.
* A result of zero indicates no relationship at all.

**Sampling Distributions**

The distribution of sample means is defined as the set of means from all the possible random samples of a specific size (n) selected from a specific set. A sampling distribution is a graph of a statistic for your sample data. While, technically, you could choose any statistic to paint a picture, some common ones you’ll come across are:

1. Mean
2. Median
3. Mode
4. Standard Deviation
5. Variance
6. Range

Let’s take a sample of size n and we will calculate the statistic value(could be either mean, median, range, std deviation, variance) to estimate the value of the parameter.

Lets take a mean of population having 3 different samples.

Now mean would be certainly 1+2+3 = 6/3 = 2

Further we can also check the associated means by taking different samples in order to have an estimated value of the parameter.

**Regression**

A regression is a statistical technique that relates a dependent variable to one or more independent (explanatory) variables. A regression model is able to show whether changes observed in the dependent variable are associated with changes in one or more of the explanatory variables.

Linear Regression: A linear regression is where the relationships between your variables can be described with a straight line. Non-linear regressions produce curved lines

It is the most widely used statistical technique; it is a way to model a relationship between two sets of variables. The result is a linear regression equation that can be used to make predictions about data.

y’ = a + bx (b = slope of the line, a is the intercept. Also denoted as b0 and b1.

Please refer to the image below for an explanation on how it works.

A diagram of a linear regression

Description automatically generated

Multiple Linear Regression: Multiple linear regression is a regression model that estimates the relationship between a quantitative dependent variable and two or more independent variables using a straight line.

Refer to the image below

A graph of multiple linear regression

Description automatically generated

Polynomial Regression: Polynomial regression, abbreviated E(y |x), describes the fitting of a nonlinear relationship between the value of x and the conditional mean of y. It usually corresponded to the least-squares method.

A line graph with blue dots and white text

Description automatically generated

**Goodness of fit**

The goodness of fit test is used to test if sample data fits a distribution from a certain population. In other words, it tells you if your sample data represents the data you would expect to find in the actual population. Goodness of fit tests commonly used in statistics are:

* Chi – Squared Test
* Kolmogorov-Smirnov.
* Anderson-Darling.
* Shipiro-Wilk.

The subscript “c” is the degrees of freedom. “O” is your observed value and E is your expected value.

To interpret the test, you’ll need to choose an alpha level (1%, 5% and 10% are common). The chi-square test will return a p-value. If the p-value is small (less than the significance level), you can reject the null hypothesis that the data comes from the specified distribution.

Since p < 0.05 is enough to reject the null hypothesis (no association), p = 0.002 reinforce that rejection only. If the significance value that is p-value associated with chi-square statistics is 0.002, there is very strong evidence of rejecting the null hypothesis of no fit. It means good fit.

**What is a Python Library?**

There are many libraries, AKA modules, and each has a specific purpose. A library is essentially a bank of reusable code. It is a set of features and/or custom data types. Modules are meant to be imported and used by your programs. . Once code has been written, it can be packaged with rules. Once the pieces of code are in standard form and directly layout, we call it a package. Modules that provide related functionality can be grouped together in a package.

A Python Library is a collection of functions and methods that can be called upon without needing to write code. Each library houses built-in modules that provide different functionalities; all functionalities can be directly used.

Python Data Analysis Libraries have Three Groups:

* Scientific Computing Libraries
* Data Visualization
* Algorithmic Libraries

**Lets learn a little more about these libraries:**

Scipy: SciPy is a collection of mathematical algorithms and convenience functions built on the NumPy extension of Python. It adds significant power to the interactive Python session by providing the user with high-level commands and classes for manipulating and visualizing data. Extensively used for high-level computations.

NumPy: NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Pandas: Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays. As one of the most popular data wrangling packages, Pandas works well with many other data science modules inside the Python ecosystem, and is typically included in every Python distribution

Matplotlib:

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. It can:

* Create publication quality plots.
* Make interactive figures that can zoom, pan, update.
* Customize visual style and layout.
* Export to many file formats.
* Embed in JupyterLab and Graphical User Interfaces.
* Use a rich array of third-party packages built on Matplotlib.

Seaborn: Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

SciKitlearn: Scikit-learn is an open source data analysis library, and the gold standard for Machine Learning (ML) in the Python ecosystem. Key concepts and features include:

* Algorithmic decision-making methods, including:
  + Classification: identifying and categorizing data based on patterns.
  + Regression: predicting or projecting data values based on the average mean of existing and planned data.
  + Clustering: automatic grouping of similar data into datasets.
* Algorithms that support predictive analysis ranging from simple linear regression to neural network pattern recognition.
* Interoperability with NumPy, pandas, and matplotlib libraries.

Statsmodels: Statsmodels is a Python library built specifically for statistics. Statsmodels is built on top of NumPy, SciPy, and matplotlib, but it contains more advanced functions for statistical testing and modeling that you won't find in numerical libraries like NumPy or SciPy. It can be used for:

* Linear Regression
* Multiple Linear Regression
* Logistics Regression
* Time Series Analysis
* Statistical Tests

**Data Analysis using the Libraries**

* There are a few standard steps one needs to take when using the Python Libraries for Data Analysis:
* Data acquisition: gather data from multiple sources- web servers(web scraping), Logs (text), databases, API, online repositories
* Data preprocessing:  data cleaning (different format, missing values, duplicates values), data transformation (normalizing
* Data modeling: build different models and evaluate them to find the best performed one
* Visualization and communication
* Deploy and maintain the model

A diagram of data analysis

Description automatically generated

**Let's Practice**

In the past lesson, we learnt about the different Python libraries and how they can be used for data analysis. We will practice a few basic problems using the Python libraries here.

**How to:**

You can use the Python editor of your choice but it is highly encouraged that you use Google Colab.

The exercise sheets and the Python notebooks for the Pandas exercises are attached below:

A white background with black text

Description automatically generated

The instructions and data sets for the Import/Export Data activity are attached below:

A white background with a black border

Description automatically generatedA screenshot of a computer

Description automatically generated

The instructions, dataset and the Python Notebook for the GoodBelly Activity are attached below:

A screenshot of a computer

Description automatically generated

DATA WRANGLING

**Data Formatting**

Data format is the definition of the structure of data within a database or file system that gives the information its meaning. Structured data is usually defined by rows and columns, where columns represent different fields corresponding to, for example, name, address, and phone number, and each field has a defined type, such as integers, floating point numbers, characters, and Boolean. Rows then represent individual records that fill in each column with its corresponding value.

Why is Data Formatting Important?

Source data can come in many different data formats. To run analytics effectively, a data scientist must first convert that source data to a common format for each model to process. With many different data sources and different analytic routines, that data wrangling can take 80 to 90 percent of the time spent on developing a new model. Having a model-driven architecture that simplifies the conversion of the source data to a standard, easy-to-use format ready for analytics reduces the overall time required and allows the data scientist to focus on machine learning model development and the training life cycle.

**Data Normalization**

Data normalization is generally considered the development of clean data. Simply put, this process includes eliminating unstructured data and redundancy (duplicates) in order to ensure logical data storage. When data normalization is done correctly, you will end up with standardized information entry.

The main objective of database normalization is to eliminate redundant data, minimize data modification errors, and simplify the query process. It also helps in keeping data consistent, allows for fair comparisons between different features and is important for data computation.

Ultimately, normalization goes beyond simply standardizing data, and can even improve workflow, increase security, and lessen costs.

There are different ways to normalize data:

**Simple Feature Scaling** is a method used to normalize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step.  Scaling can make a difference between a weak machine learning model and a better one.

Feature scaling is essential for machine learning algorithms that calculate distances between data. If not scale, the feature with a higher value range starts dominating when calculating distances.

Additional information found [here](https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35)

**Min-max** normalization is one of the most common ways to normalize data. For every feature, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1, and every other value gets transformed into a decimal between 0 and 1.

Min-max normalization has one fairly significant downside: it does not handle outliers very well. For example, if you have 99 values between 0 and 40, and one value is 100, then the 99 values will all be transformed to a value between 0 and 0.4. That data is just as squished as before!

Min-max normalization using python found [here](https://towardsdatascience.com/everything-you-need-to-know-about-min-max-normalization-in-python-b79592732b79)

**Z-score** normalization is a strategy of normalizing data that avoids this outlier issue. The formula for Z-score normalization is below:

                                                                    value−μ/σ

Here, μ is the mean value of the feature and σ is the standard deviation of the feature. If a value is exactly equal to the mean of all the values of the feature, it will be normalized to 0. If it is below the mean, it will be a negative number, and if it is above the mean it will be a positive number. The size of those negative and positive numbers is determined by the standard deviation of the original feature. If the unnormalized data had a large standard deviation, the normalized values will be closer to 0.

Additional information found [here](https://www.codecademy.com/article/normalization)

**Data Binning**

Data binning, bucketing is a data pre-processing method used to minimize the effects of small observation errors. The original data values are divided into small intervals known as bins and then they are replaced by a general value calculated for that bin. This has a smoothing effect on the input data and may also reduce the chances of overfitting in the case of small datasets  
There are 2 methods of dividing data into bins:

1. Equal Frequency Binning: bins have an equal frequency.
2. Equal Width Binning : bins have equal width with a range of each bin are defined as [min + w], [min + 2w] …. [min + nw] where w = (max – min) / (no of bins).

**Let's Practice**

In the past lesson, we learnt about the steps that are a part of Data pre-processing that involved Data Formatting and Data normalization. We learnt why these steps are important and how they make the crude data more insightful. Here, we will apply that learning with a few practice problems.

**How to:**

You can use the Python editor of your choice but it is highly encouraged that you use Google Colab.

The exercise sheets and the instructions for Data Normalization  are attached below

A screenshot of a computer

Description automatically generated

The following datasheet is an example of crude data. Recollecting all the steps of Data Formatting, format the following data. Try Binning and creating Histograms on the following set of the data as well.

A close-up of a white background

Description automatically generated

The following data comprises the number of Covid cases around the world and their recovery rate. Download the data attached and then try solving the problem questions using Google Colab from the Activity instructions attached.

A screenshot of a computer

Description automatically generated

**Exploratory Data Analysis**

Exploratory data analysis (EDA) is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling and thereby contrasts traditional hypothesis testing.

Exploratory data analysis has been promoted by John Tukey since 1970 to encourage statisticians to explore the data, and possibly formulate hypotheses that could lead to new data collection and experiments. EDA is different from initial data analysis (IDA), which focuses more narrowly on checking assumptions required for model fitting and hypothesis testing, and handling missing values and making transformations of variables as needed. EDA encompasses IDA.

**What are Descriptive Statistics?**

Descriptive statistics are brief informational coefficients that summarize a given data set, which can be either a representation of the entire population or a sample of a population. Descriptive statistics are broken down into measures of central tendency and measures of variability (spread).

Descriptive statistics, in short, help describe and understand the features of a specific data set by giving short summaries about the sample and measures of the data.

You can summarize statistics using pandas describe() method.

You can summarize the categorical data by using the value\_counts() method.

Box plot 🡪method for graphically depicting groups of numerical data through their quartiles: minimum, first quartile, median, third quartile, and maximum.

Scatter plot 🡪 uses dots to represent values for two different numeric variables. They are used to observe relationships between variables.  There are 3 possible correlations: positive, negative, and no correlation.

Let's quickly try something. Below are a few random data points:

35, 29, 34, 25, 29, 28, 38, 37, 35, 30

What are the steps that you would take prepare the data to be visualized in a box plot?

**GroupBy**

GroupBy is used for grouping the data according to the categories and applying a function to the categories. It also helps to aggregate data efficiently. The Pandas groupby() is a very powerful function with a lot of variations. It makes the task of splitting the Dataframe over some criteria really easy and efficient. The syntax and the parameters are as follows:

DataFrame.groupby(by=None, axis=0, level=None, as\_index=True, sort=True, group\_keys=True, squeeze=False, \*\*kwargs)

Parameters :

* by : mapping, function, str, or iterable
* axis : int, default 0
* level : If the axis is a MultiIndex (hierarchical), group by a particular level or levels
* as\_index : For aggregated output, return object with group labels as the index. Only relevant for DataFrame input. as\_index=False is effectively “SQL-style” grouped output
* sort : Sort group keys. Get better performance by turning this off. Note this does not influence the order of observations within each group. groupby preserves the order of rows within each group.
* group\_keys : When calling apply, add group keys to index to identify pieces
* squeeze : Reduce the dimensionality of the return type if possible, otherwise return a consistent type

**Correlation**

Correlation is a statistical metric for measuring to what extend variables are interdependent. Correlation DOES NOT IMPLY causation. Correlation can result into:

* Positive linear relationship
* Negative linear relationship
* No relationship

**Chi Square Test**

This statistical test shows a relationship between two categorical variables. It does NOT tell you what kind of relationship exists between both variables; it only indicates whether there is a relationship.

Watch this [video(opens in a new tab)](https://youtu.be/7_cs1YlZoug) to know more.

**Let's Practice**

In the past lesson, we learnt about exploratory data analysis and about the various statistical measures. We will practice EDA using Pandas in this lesson.

**How to:**

You can use the Python editor of your choice but it is highly encouraged that you use Google Colab.

The datasheet and the instructions for the Exploratory Data analysis are attached below:

A screenshot of a computer

Description automatically generated

**What is a Model?**

A model is simply a system for mapping inputs to outputs.

For example, if we want to predict house prices, we could make a model that takes in the square footage of a house and outputs a price.

A model represents a theory about a problem: there is some connection between the square footage and the price and we make a model to learn that relationship. Models are useful because we can use them to predict the values of outputs for new data points given the inputs.

**What is model development or model fitting?**

Model fitting is a measure of how well a machine learning model generalizes to similar data to that on which it was trained.

A model that is well-fitted produces more accurate outcomes.

**Simple Linear Regression**

A statistical method that allows us to summarize and study relationships between two continuous (quantitative) variables:

One variable, denoted x, is regarded as the predictor, explanatory, or independent variable.

The other variable, denoted y, is regarded as the response, outcome, or dependent variable.

Linear Regression Model tries and explains the statistical relationship!

Y^ = b0+b1x1

Where:

y^ = is the predicted response (or fitted value) for experimental unit

Y1 = denotes the observed response for experimental unit i

X1 =  denotes the predictor value for experimental unit i

B0 = Intercept of the line

B1 = slope of the line

**Let's try something**

Following is a dataset that represent two variables X being TV and Y being Sales.

Use this data in the table below to create a scatter plot and find the best fitting line that runs through it! (You can use Excel to do it) Once you have the line, cross check with your peers if they have the same line as you, discuss if it is any different!

|  |  |
| --- | --- |
| X (TV) | Y (Sales) |
| 230.1 | 22.1 |
| 44.5 | 10.4 |
| 17.2 | 9.3 |
| 151.5 | 18.5 |
| 180.8 | 12.9 |
| 38.2 | 7.6 |
| 94.2 | 9.7 |

**Do you recollect?**

In the introductory Statistics course, do you remember what Multiple Linear Regression and Polynomial Linear Regression is? And when and why are they used?

A diagram of a process

Description automatically generated

**Let's Practice**

In the past lesson, we learnt about Data Modeling and how it helps in machine learning and decision making. We will practice a few problems here.

**How to:**

You can use the Python editor of your choice but it is highly encouraged that you use Google Colab.

The exercise sheets and the Python notebooks for the Data Modeling exercises are attached below:

A screenshot of a computer

Description automatically generated

MODEL EVAUATION & REFINEMENT

**What is test data, train data and validation data?**

Training Dataset: The sample of data used to fit the model.

Validation Dataset: The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.

Test Dataset: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

**Split Data**

A screenshot of a computer

Description automatically generated

The syntax used for Data split is as follows:

train\_test\_split()

X\_train,x\_test,y\_train,y\_test = train\_test\_split(x\_data, y\_data, test\_size = 0.3, randome\_state =0)

**Cross Validation**

Cross-validation, sometimes called rotation estimation or out-of-sample testing, is any of various similar model validation techniques for assessing how the results of a statistical analysis will generalize to an independent data set.

Cross-validation is a resampling method that uses different portions of the data to test and train a model on different iterations. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

In a prediction problem, a model is usually given a dataset of known data on which training is run (training dataset), and a dataset of unknown data (or first seen data) against which the model is tested (called the validation dataset or testing set).

The goal of cross-validation is to test the model's ability to predict new data that was not used in estimating it, in order to flag problems like overfitting or selection bias[10] and to give an insight on how the model will generalize to an independent dataset (i.e., an unknown dataset, for instance from a real problem).

An example of the syntax for the same is below:

Scores = cross\_val\_score(lr, x\_data, y\_data, cv =3)

np.mean(scores)

Yhat = cross\_val\_predict(lr2e, x\_data, y\_data, cv = 3)

**Overfitting and Underfitting**

Overfitting refers to a model that models the training data too well.

Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the models ability to generalize.

Overfitting is more likely with nonparametric and nonlinear models that have more flexibility when learning a target function. As such, many nonparametric machine learning algorithms also include parameters or techniques to limit and constrain how much detail the model learns.

For example, decision trees are a nonparametric machine learning algorithm that is very flexible and is subject to overfitting training data. This problem can be addressed by pruning a tree after it has learned in order to remove some of the detail it has picked up.

Underfitting refers to a model that can neither model the training data nor generalize to new data.

An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.

Underfitting is often not discussed as it is easy to detect given a good performance metric. The remedy is to move on and try alternate machine learning algorithms. Nevertheless, it does provide a good contrast to the problem of overfitting.

A diagram of a graph

Description automatically generated

**Ridge Regression**

Ridge regression is the method used for the analysis of multicollinearity in multiple regression data. It is most suitable when a data set contains a higher number of predictor variables than the number of observations. The second-best scenario is when multicollinearity is experienced in a set.

Multicollinearity happens when predictor variables exhibit a correlation among themselves. Ridge regression aims at reducing the standard error by adding some bias in the estimates of the regression. The reduction of the standard error in regression estimates significantly increases the reliability of the estimates.

A diagram of a diagram of a blue circle and a green circle

Description automatically generated

In summary, Ridge regression will prevent overfitting. In Python, you can use the following syntax:

From sklearn.linear-model import Ridge

RidgeModel = Ridge(alpha=0.1)

RidgeModel.fit(X,Y)

Yhat = RidgeModel.predict(Y)

**Let's Practice**

In the past lesson, we learnt about model refinement. We learnt how to develop a model and how to use it to train the data. We learnt about underfitting, overfitting and the causes and the remedies for the same. We also learnt what Ridge Regression is and when to use it.

**How to:**

You can use the Python editor of your choice but it is highly encouraged that you use Google Colab.

The dataset and the instructions can be found in the file attached below:

A white rectangular object with black border

Description automatically generated