Salary Estimation using K-Nearest Neighbour

1. Finding the Problem - Application
- **To Predict**: Predicting whether this Job applicant got Salary above 50K or Not from Previous Company - HR

2. Collecting Dataset
- **Input: Age, Education number, Capital Gain & Hours/week Output: Salary above/below 50K**: Based on Age, Education Number, Capital Gain, Hours per week to estimate the salary is above 50K or below 50K

3. Load & Summarize Dataset — Load Dataset from the directory & Summarize the details such as no. of rows and Columns & Content
- **Pandas - Load CSV Format Dataset**: dataset = pandas.read_csv('dataset.csv')
- **No. of Rows & Columns**: dataset.shape
- **Display 1st 5 rows of dataset**: dataset.head(5)

4. Mapping Data from to Text to Binary Numbers
- **Function: .map**: If the data is <=50K or >=50K, kind of text. Here we need to Map <50K as 0 & >50K as 1

5. Segregating Dataset into X & Y — iloc - It helps us select a value that belongs to a particular row or column
- **SYNTAX: dataset.iloc[:, start_col:end_col]**
  - X = dataset.iloc[:, :-1].values
  - Y = dataset.iloc[:, -1].values

6. Splitting Dataset to Train & Test — Useful for validation
- train_test_split(X, Y, test_size = 0.25, random_state = 0)

7. Feature Scaling
- **PROBLEM**: Since both the features have different scales, there is a chance that higher weightage is given to features with higher magnitude. This will impact the performance of the machine learning algorithm and obviously, we do not want our algorithm to be biassed towards one feature.
- **SOLUTION**: we scale our data to make all the features contribute equally to the result

Name	Weight	Price
Orange	15	1
Apple	18	3
Banana	12	2
Grape	10	5

- **Types**:

-1.184341	1.520013
-1.184341	-1.100699
0.416120	-1.100699
1.216350	0.209657
0.736212	0.471728

  - **Standardization**: $X' = \dfrac{X - \mu}{\sigma}$ — Here the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.
  - **Normalization**: $X' = \dfrac{X - X_{min}}{X_{max} - X_{min}}$ — Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling

8. Algorithm — K-Nearest Neighbor

- **Definition**:
  - KNN makes predictions using the training dataset directly. Predictions are made for a new data point by searching through the entire training set for the K most similar instances (the neighbors) and summarizing the output variable for those K instances.
  - Lazy Learning: No learning of the model is required and all of the work happens at the time a prediction is requested.
  - KNN can be used for regression and classification problems.
  - When KNN is used for regression problems the prediction is based on the mean or the median of the K-most similar instances.

- Based on Minkowski Distance Metric we gonna classify the data points |
  p = 1 , Manhattan Distance
  p = 2 , Euclidean Distance

  $$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$

  - **Minkowski distance - Generalization of Euclidean and Manhattan distance.**
    - **Euclidean Distance**: $\sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$ — Euclidean distance is calculated as the square root of the sum of the squared differences between a new point (x) and an existing point (y) — Euclidean is a good distance measure to use if the input variables are similar in type (e.g. all measured widths and heights).
    - **Manhattan Distance**: $\sum_{i=1}^{k} |x_i - y_i|$ — This is the distance between real vectors using the sum of their absolute difference. Also called City Block Distance. — Manhattan distance is a good measure to use if the input variables are not similar in type (such as age, gender, height, etc.).
  - **Hamming Distance**: $D_H = \sum_{i=1}^{k} |x_i - y_i|$ — It is used for categorical variables. If the value (x) and the value (y) are the same, the distance D will be equal to 0 . Otherwise D=1

- **Advantage**:
  - Easy to use
  - Quick Calculation time
  - Does not make assumptions about the data

- **Disadvantage**:
  - Distance of the new instance is computed with all the training samples every time
  - Choosing the value of K is a tough choice to make as lower value of K results in higher influence of noise whereas increasing the number of K might result into overfitting.
  - KNN is computationally expensive as it searches the nearest neighbors for the new point at the prediction stage

- **Application**:
  - Movie and product recommendation system
  - Text mining
  - Medicine
  - Stock prediction
  - Climate forecasting

9. Finding the Best K-Value - number of neighbors
- Choose the K Value where we are getting least mean error

- From this figure, we can observe that K Value range from 15 to 35, our mean error is low

10. Training — Training our Model for Pre-processed Dataset
- model.fit(X_train, y_train)

11. Validation — Obtaining the accuracy of the Model — **Confusion Matrix**

12. Prediction — Observing how our model is classifying our new data
- result = model.predict(sc.transform(newEmp))