## Week 15 Data Science Review Document

**Overview**

You are now in Week 15 of a 35-week Data Science journey. This week focuses on three core areas:

1. **Data Pre-processing and EDA**
2. **Git Concepts**
3. **Machine Learning Concepts**

Additionally, you’ll revisit several previously pending foundational topics. The document is structured to provide detailed explanations, common pitfalls, and hands-on tasks.

**1. Data Pre-processing & Exploratory Data Analysis (EDA)**

**Concepts to Learn:**

* **Handling missing values (Imputation)**
* **Feature scaling (Normalization & Standardization)**
* **Outlier detection (Box Plot, IQR)**
* **Data Transformation (e.g., Log Transformation)**
* **Insight extraction**

**Tasks:**

* Select **3 new datasets** (not previously used)
* Perform:
  + Missing value treatment
  + Outlier detection using Box Plots & IQR
  + Log transformation if needed
  + Normalization/Standardization
  + Extract and document insights

**Common Issues:**

* Forgetting to check for nulls in each column
* Not justifying why transformation (like log) was used
* Poor visualization or interpretation

**Example Tools/Libraries:**

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler, MinMaxScaler

**Notes:**

* **Imputation**: Use fillna() or SimpleImputer
* **Box Plot** helps identify outliers visually
* **IQR Method**: Q3 - Q1, remove rows outside 1.5 \* IQR
* **Log Transformation**: Use when dealing with skewed distributions
* **Normalization** is good for distance-based models (KNN, KMeans)

**2. Git Concepts**

**Concepts to Learn:**

* Git init, add, commit, push, pull
* Branching
* git rebase, git reset, git log

**Task:**

* Create a GitHub account (if not already done)
* Initialize a repo, push all Week 15 tasks
* Practice branching and merging

**Common Issues:**

* Forgetting to git add before commit
* Merge conflicts when not pulling latest code

**Example Commands:**

git init

git add .

git commit -m "Week 15 task upload"

git branch -M main

git remote add origin <repo-url>

git push -u origin main

**Git Rebase vs Reset:**

* rebase integrates changes in a cleaner history (linear)
* reset can undo commits (be careful - --hard is destructive)

**3. Machine Learning Concepts**

**Topics:**

* **Vectorization**: Replacing for-loops with NumPy operations for speed
* **Performance**: Measuring with accuracy, precision, recall, F1-score
* **Optimization**: Gradient Descent, Cost Functions

**Task:**

* Select **1 new dataset**
* Perform:
  + EDA
  + Preprocessing
  + Modeling (any basic model: Linear Regression, Logistic Regression)
  + Apply vectorization if manual calculations are involved

**Vectorization:**

import numpy as np

X = np.array([1, 2, 3])

W = np.array([0.4, 0.5, 0.6])

result = np.dot(X, W)

**Gradient Descent:**

def gradient\_descent(x, y, lr=0.01, epochs=100):

m, b = 0, 0

for \_ in range(epochs):

y\_pred = m \* x + b

error = y - y\_pred

m\_grad = -2 \* np.dot(x, error) / len(x)

b\_grad = -2 \* np.sum(error) / len(x)

m -= lr \* m\_grad

b -= lr \* b\_grad

return m, b

**Common Issues:**

* Using loops instead of NumPy or pandas vectorized operations
* Misunderstanding shape mismatches in dot products

**Pending Topics Overview**

**Imputation:**

Filling missing data using:

* Mean, median, or mode
* Predictive models

**Gradient Descent:**

Used to minimize cost/loss function by updating parameters in the direction of the negative gradient.

**Vectorization:**

Faster computation using NumPy arrays instead of loops.

**Box Plot, IQR:**

Used to detect outliers and understand data spread.

**Log Transformation:**

Reduces right-skewed distributions to make data normal.

**Normalization:**

Brings all features to the same scale (0 to 1), useful in distance-based models.

**Git Commands:**

* git log: view commit history
* git reset: undo changes
* git rebase: integrate changes from one branch to another in a linear fashion

**PCA (Principal Component Analysis):**

Used for **Dimensionality Reduction** by transforming original features into fewer components that retain maximum variance.

**Final Deliverables**

* EDA report for 3 datasets with visualizations
* GitHub repo with pushed code and README
* ML model with clear comments explaining vectorization, performance, and optimization
* A summary document reflecting insights, opinions, and concepts learned

**Tips for Success**

* Document every step while doing the EDA and ML tasks
* Use Jupyter Notebooks or Markdown cells to explain code
* Regularly commit code with clear messages
* Refer to official documentation (pandas, sklearn, git-scm.com)

## This is How I pushed my code to already existing git hub :

### ✅ Steps to Add and Push Your New Folder via CMD

Assuming you're already inside your project folder:

cd path\to\Machine-Learning

### 🔹 1. ****Check Git status****

git status

This shows you the new untracked EDA/ folder and its files.

### 🔹 2. ****Stage the new folder****

git add notebooks/EDA

Or to stage **everything**, just:

git add .

### 🔹 3. ****Commit the changes****

git commit -m "Added EDA folder with analysis notebooks"

### 🔹 4. ****Push to GitHub****

If you're already connected to a remote GitHub repo:

git push origin main

(replace *main* with *master* or your branch name if different)

### 💡 Optional: If You Haven’t Set the Remote Yet

git remote add origin https://github.com/your-username/Machine-Learning.git

git push -u origin main

### 🧠 Recap of All Commands:

cd path\to\Machine-Learning

git status

git add .

git commit -m "Added EDA folder with analysis notebooks"

git push origin main

Let me know if you want a shortcut script or you're getting any errors during push!