# AIDS-I Assignment No 2

**Q.1 Use the following data set for question 1**

82, 66, 70, 59, 90, 78, 76, 95, 99, 84, 88, 76, 82, 81, 91, 64, 79, 76, 85, 90

1. Find the Mean
2. Find the Median
3. Find the Mode
4. Find the Interquartile range

ANS:

1. **Mean**: The mean is the average.
   Sum = 82 + 66 + 70 + 59 + 90 + 78 + 76 + 95 + 99 + 84 + 88 + 76 + 82 + 81 + 91 + 64 + 79 + 76 + 85 + 90 = 1611
   Number of values = 20
   Mean= 1611 / 20
   = **80.55**
2. **Median:** The median is the middle value when the data is sorted
   Sorted data: 59, 64, 66, 70, 76, 76, 76, 78, 79, 81, 82, 82, 84, 85, 88, 90, 90, 91, 95, 99
   Median = **81.5**
3. **Mode:** The mode is the number that appears the most.
   Mode = **76**
4. **Interquartile Range**
   IQR = Q3 - Q1
   Q1 = median of the first half = 76
   Q3 = median of the second half = 89
   IQR = 89 - 76
   = **13**

**Q.2 1) Machine Learning for Kids 2) Teachable Machine**

ANS: Tool: **Machine Learning for Kids**

1. **Target Audience**
   The primary target audience for Machine Learning for Kids includes young students aged 8–17, beginners with no prior exposure to machine learning, and educators seeking to introduce foundational AI and machine learning concepts in a classroom setting.

2. **Use of this Tool by the Target Audience:**
   Machine Learning for Kids allows students to create and train simple machine

learning models by labeling datasets and using them in interactive projects. Students can train models to recognize text, images, numbers, or sounds, and subsequently deploy these models in practical applications such as games, chatbots, or mobile applications using Scratch, Python, or App Inventor.
For example, students can train a model to differentiate between images of cats and dogs and then use this model within a Scratch project to classify new images**.**

3. **Tool's Benefits and Drawbacks**

| **Benefits** | **Drawbacks** |
|---|---|
| Highly user-friendly interface that enables students to engage with machine learning concepts without requiring advanced programming skills. | Limited scalability; the platform is not designed for handling large datasets or complex deep learning models |
| Provides practical, hands-on experience with real-world machine learning workflows. | Primarily supports basic models, making it unsuitable for more sophisticated machine learning projects. |
| Seamlessly integrates with platforms such as Scratch and App Inventor, facilitating interactive project development. | May oversimplify fundamental machine learning principles, which could be less effective for older or more advanced learners. |
| Freely available and easily accessible online, making it ideal for educational use. | Certain functionalities may be dependent on stable internet connectivity. |

4. **Predictive Analytic:**
The platform is primarily designed for building models that predict outcomes based on previously labeled data. For instance, after training on examples of positive and negative reviews, a model can predict the sentiment of a new review. This predictive nature classifies the tool under predictive analytics.

5. **Supervised Learning**
Machine Learning for Kids uses labeled data during the model training phase, where each input is paired with a corresponding output. Students manually label their data (e.g., images tagged as "cat" or "dog") and the system learns to classify new, unseen data based on these labels. This approach is characteristic of supervised learning.

Tool: **Teachable Machine**

1. **Target Audience**
   The primary target audience for Teachable Machine includes students, educators, hobbyists, and beginners who wish to experiment with machine learning without requiring advanced programming skills. It is particularly suitable for individuals aged 12 and above who are interested in quickly building and deploying machine learning models in a user-friendly environment.

2. **Use of the Tool by the Target Audience**
   Teachable Machine enables users to create machine learning models by providing examples and training the model directly within a web interface. Users can train models to recognize images, sounds, or poses through simple drag-and-drop actions and webcam recordings.
   For instance, a student could train an image classification model by showing different objects to a webcam and labeling them, then export the trained model for use in websites, applications, or devices — all without writing a single line of code.

3. **Tool's Benefits and Drawbacks**

| **Benefits** | **Drawbacks** |
|---|---|
| Extremely easy to use, requiring no prior coding knowledge. | Limited model complexity, making it unsuitable for professional or advanced applications. |
| Supports quick prototyping and instant feedback, allowing users to iterate rapidly. | Cannot fine-tune model parameters, limiting customization and optimization |
| Provides export options to TensorFlow.js, TensorFlow Lite, and other formats, enabling integration into external applications | Models are trained locally in-browser, which can lead to performance constraints on devices with lower processing power. |
| Free to use and accessible online, promoting widespread experimentation with AI. | Not ideal for large-scale datasets or sophisticated multi-class problems. |

4. **Predictive Analytic**
   Teachable Machine builds models that predict outcomes based on user-provided examples. For example, after training a model with images of different emotions (happy, sad, surprised), the model can predict the emotion shown in a new image. This predictive functionality categorizes Teachable Machine under predictive analytics.

5. **Supervised Learning**
   Teachable Machine relies on labeled data provided by the user during the training process. Users must supply categorized examples (e.g., images labeled "cat," "dog," "car") to teach the model how to classify new inputs. This process follows the principles of supervised learning, where the system learns a mapping from inputs to known outputs.

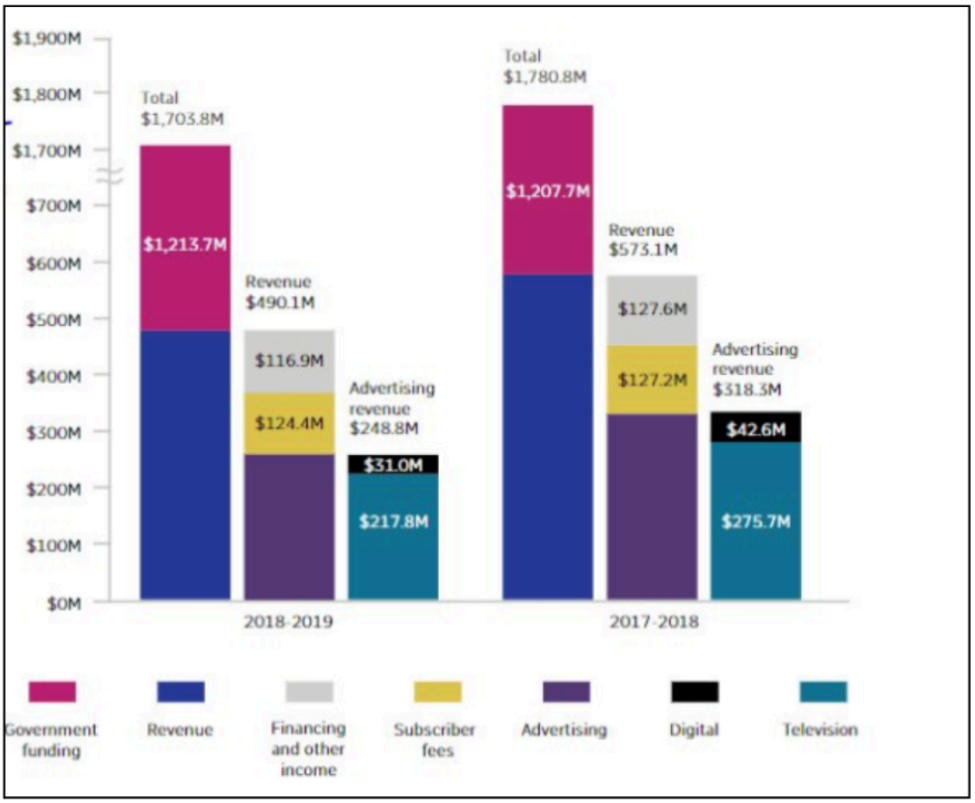
**Q.3 Data Visualization**

Sources:
https://thedeepdive.ca/cbc-commits-a-chart-crime-in-representing-its-funding-and-revenue-sources/

https://www.codeconquest.com/blog/12-bad-data-visualization-examples-explained/#toc-bad-data-visualization-example-2

Misleading CBC(Canadian national broadcast company) Funding Bar Chart:

In April 2023, a bar chart from the Canadian Broadcasting Corporation's (CBC) 2018–2019 Annual Report resurfaced online and quickly drew widespread criticism for its misleading design. The chart was intended to show CBC's funding sources, including government appropriations and advertising revenue. However, it sparked controversy after viewers noticed serious issues with how the data was presented.



The axis had a sudden break—jumping from $700M to $1.7B—causing a $490M revenue bar to appear visually larger than the $1.2B government funding bar. This misleading scale made it seem like television revenue equaled or exceeded government funding, which is factually incorrect. Additionally, the chart layout was flawed: revenue and advertising revenue were shown as separate bars rather than subdivisions of the total income, creating further confusion.

**Q. 4 Train Classification Model and visualize the prediction performance of trained model required information**

ANS: **Model Used: SVM**

1. Import required libraries

```
[1]  # Import necessary libraries
     import pandas as pd
     import numpy as np
     from sklearn.model_selection import train_test_split, GridSearchCV
     from sklearn.preprocessing import StandardScaler
     from sklearn.svm import SVC
     from sklearn.metrics import classification_report, confusion_matrix, ConfusionMatrixDisplay, accuracy_score
     from imblearn.over_sampling import SMOTE
     import matplotlib.pyplot as plt
     import seaborn as sns
```

2. Loading the dataset

```
[2]  data = pd.read_csv('/content/sample_data/diabetes.csv')
```

3. Performing data preprocessing

| | 0 |
|---|---|
| Pregnancies | 0 |
| Glucose | 0 |
| BloodPressure | 0 |
| SkinThickness | 0 |
| Insulin | 0 |
| BMI | 0 |
| DiabetesPedigreeFunction | 0 |
| Age | 0 |
| Outcome | 0 |

Since there are no null values, we can skip imputation.

4. Standardization of data

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Feature scaling standardizes the values of your features so they all have: Mean = 0 and Standard Deviation = 1

5. Resolve Class Imbalance using SMOTE

```
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_scaled, y)
```

SMOTE (Synthetic Minority Over-sampling Technique) is used to fix class imbalance by creating synthetic samples for the minority class.

6. Train, Validation, Test Split

```
# Train, Validation, Test Split
X_train_full, X_test, y_train_full, y_test = train_test_split(
    X_resampled, y_resampled, test_size=0.10, random_state=42, stratify=y_resampled
)
X_train, X_val, y_train, y_val = train_test_split(
    X_train_full, y_train_full, test_size=2/9, random_state=42, stratify=y_train_full
)  # 2/9 to split 70/20/10 correctly
```

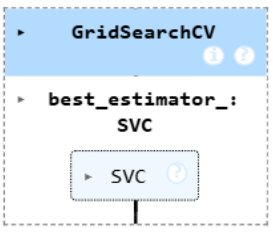
7. Define the model

```
[8]  svm_model = SVC()
```

8. Hyperparameter Tuning using GridSearchCV

```
# Hyperparameter Tuning using GridSearchCV
param_grid = {
    'C': [0.1, 1, 10, 100],
    'gamma': [1, 0.1, 0.01, 0.001],
    'kernel': ['rbf']
}

grid = GridSearchCV(svm_model, param_grid, refit=True, verbose=0, cv=5)
grid.fit(X_train, y_train)
```



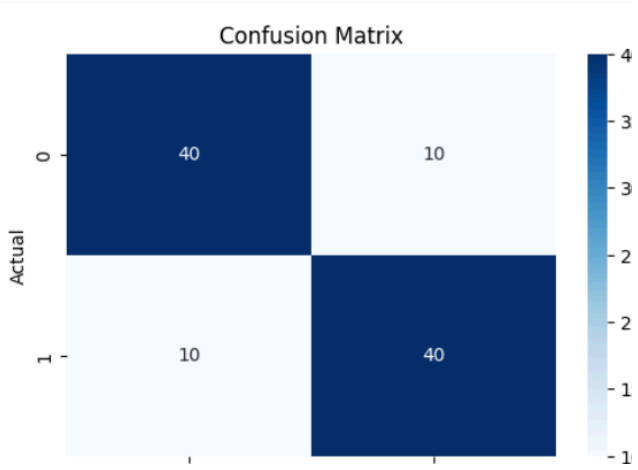
9. Evaluate on Validation set

```
Validation Accuracy: 0.825
```

10. Evaluation on Test Set

```
Test Accuracy: 0.8
```

11. Evaluating the model




**Validation Accuracy**: **82.5%**
The model achieved an accuracy of **82.5%** on the validation set, indicating good generalization during training.
**Test Accuracy**: **80%**
When evaluated on the unseen test set, the model obtained an accuracy of **80%**, showing consistent performance and minimal overfitting.

**Confusion Matrix Analysis**:

a. **True Positives (TP)**: 40
b. **True Negatives (TN)**: 40
c. **False Positives (FP)**: 10
d. **False Negatives (FN)**: 10

The confusion matrix shows a **balanced classification** performance:

- Most samples are correctly classified.
- Only 10 instances are misclassified in each class (positive and negative).
- The model is **well-generalized** and not overfitting.

**Q.5 Train Regression Model and visualize the prediction performance of trained model**

1. Importing required libraries

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import Ridge
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import seaborn as sns
```

2. Define a Regression model
   Firstly we define
   class RegressionModel:

```
def __init__(self, model, param_grid):
    self.model = model
    self.param_grid = param_grid
    self.best_model = None
```

3. Load Dataset

```
def load_data(self, file_path):
    self.data = pd.read_csv(file_path)
    self.X = self.data.iloc[:, [0]]  # First column as independent
    self.y = self.data.iloc[:, 1:5]   # Columns 2 to 5 as dependent
```

4. Split Data (70/30)

```
def split_data(self, test_size=0.3):
    self.X_train, self.X_test, self.y_train, self.y_test = train_test_split(
        self.X, self.y, test_size=test_size, random_state=42
    )
```

5. Tune the hyperparameters

```
def tune_hyperparameters(self):
    grid_search = GridSearchCV(self.model, self.param_grid, cv=5)
    grid_search.fit(self.X_train, self.y_train)
    self.best_model = grid_search.best_estimator_
```

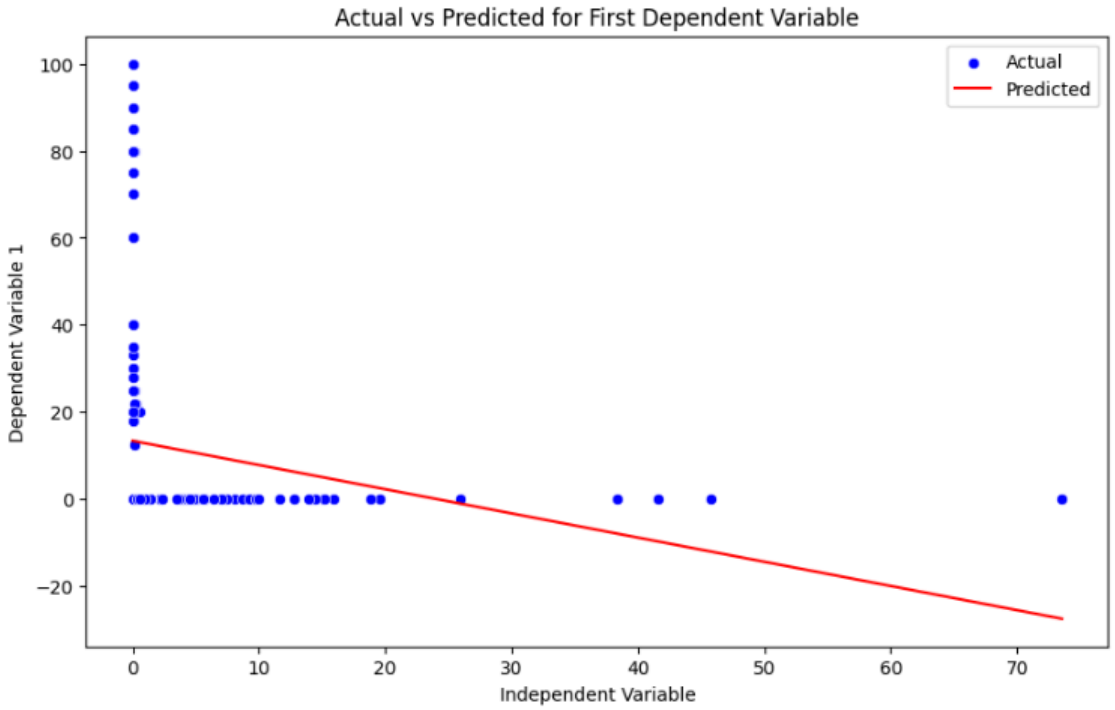
6. Train model

```
def train(self):
```

```
self.best_model.fit(self.X_train, self.y_train)
```

7. Evaluation and Visualization

```
R2 Score: 0.1069
Adjusted R2 Score: 0.1009
Mean Squared Error: 158.5635
```




The model explains approximately 11% of the variance in the target variable. While the performance is relatively low, this is expected in cases where:

- Important features might be missing,
- The relationship between features and the target is complex or non-linear,
- The model might be underfitting the data.
- The Adjusted $R^2$ is slightly lower than the $R^2$, indicating that adding more features has not significantly improved the model's performance after accounting for complexity.

**Why we May Not Reach $R^2 > 0.99$**

- Missing Features: Key factors like interior conditions, recent renovations, and market trends are not included.
- Noisy Data: Real-world housing prices are influenced by unpredictable factors, adding noise.
- Non-linear Relationships: The model may not fully capture complex, non-linear patterns in the data.

- Outliers: Extreme property values make the data harder to model accurately.
- Bias-Variance Tradeoff: Higher polynomial degrees risk overfitting, while simpler models underfit.

**Q.6** What are the key features of the wine quality data set? Discuss the importance of each feature in predicting the quality of wine? How did you handle missing data in the wine quality data set during the feature engineering process? Discuss the advantages and disadvantages of different imputation techniques. (Refer dataset from Kaggle).

ANS:
The Wine Quality dataset from Kaggle includes several key physicochemical features that influence the quality of wine, such as fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free and total sulfur dioxide, density, pH, sulphates, alcohol.

Each of these features plays a role in determining the taste, stability, and preservation of the wine. For instance, high levels of volatile acidity often lower the quality, while alcohol content and sulphates generally show a positive correlation with better wine ratings.

During the feature engineering process, handling missing data is crucial for model performance. Although the original dataset is relatively clean, if missing values are present, the following techniques can be used:

1. Remove Missing Data:
Delete rows or columns with missing values.
Best when only a small portion of data is missing.

2. Fill with Mean:
Replace missing values with the column's average.
Works well for normally distributed numerical data.

3. Fill with Median:
Use the middle value of the column for imputation.
Better than mean when data has outliers or is skewed.

5. Fill with Constant:
Replace missing data with a fixed value like 0 or "unknown."
Helps retain rows while signaling missingness.

Advantages and disadvantages of different imputation techniques

1. <u>Mean Imputation</u>

Advantage: Simple to implement and works well with normally distributed data.
Disadvantage: Sensitive to outliers and may distort the overall variance.

2. Median Imputation

Advantage: More robust than mean; not affected by outliers.
Disadvantage: Does not consider relationships between different features.

3. Mode Imputation

Advantage: Ideal for categorical data; retains the most common category.
Disadvantage: Can lead to overrepresentation of one category.

4. Constant Value Imputation

Advantage: Easy to implement and helps highlight missing data.
Disadvantage: May introduce meaningless or misleading values.