HACKATHON DOCUMENTATION

KIET-CSC-TEAM-10

KURUPUDI SIVA DURGA NAGA GOVINDU [ 246Q5A4602 ]

MOLLETI SATYANARAYANA [ 236Q1A4607 ]

GARIKE SRIDEVI [ 236Q1A4604 ]

NOKKI PRIYA DARSHINI [ 236Q1A4602 ]

KUNCHE SURYA SRI VENKATA SAI [ 236Q1A4611]

DODDI DURGA PRASAD [ 236Q1A4609 ]

DATE – 15/03/25

ACTIVITY

09:00 AM TO 10:30 AM

**GIT & GITHUB**

***GIT-***

* Git is a free, open-source, and distributed version control system.
* It's used to track changes in files and collaborate with others on projects.
* GIT is used in a offline which means it is used in local system.

***GITHUB-***

* GitHub is a web-based platform that hosts Git repositories.
* It provides tools for collaboration, code sharing, and project management**.**
* GitHub needs an internet source to access.

***DIFFERENCE BETWEEN GIT AND GITHUB***

* **Git** is a **version control system** that helps track changes in code, allowing multiple developers to collaborate efficiently. It runs locally on your computer.
* **GitHub** is a **cloud-based hosting service** for Git repositories, making it easy to store, share, and collaborate on code online.

10:30 AM TO 11:30 AM

**DATA COLLECTION & DATA PREPROCESSING**

Data collection involves gathering raw data, while data preprocessing prepares that data for analysis by cleaning, transforming, and organizing it into a usable format

***Data Collection****:*

* **Definition:** Data collection is the process of gathering and compiling information from various sources.
* **Purpose:** To acquire the raw data needed for analysis, modeling, or machine learning.
* **Methods:** Surveys, experiments, observations, databases, web scraping, and APIs.

***Data Preprocessing****:*

* **Definition:** Data preprocessing is the process of preparing raw data for analysis by cleaning, transforming, and organizing it into a usable format.
* **Purpose:** To improve data quality, ensure consistency, and make the data more manageable for analysis and modeling.
* **Key Steps:**
  + **Data Cleaning:** Addressing missing values, handling outliers, and correcting inconsistencies.
  + **Data Transformation:** Converting data into a suitable format for analysis, such as scaling, normalization, or encoding categorical variables.
  + **Data Integration:** Combining data from multiple sources.
  + **Data Reduction:** Reducing the volume of data while preserving important information.
  + **Feature Engineering:** Creating new features from existing ones.
  + **Data Splitting:** Dividing the dataset into training, validation, and testing sets.

11:30 AM TO 12:30 PM

**LINEAR & LOGISTIC REGRESSION**

Linear and logistic regression are both regression analysis techniques used to model relationships between variables, but they differ in their output: linear regression predicts continuous values, while logistic regression predicts probabilities for categorical outcomes, often binary (yes/no, 0/1).

Here's a more detailed comparison:

**Linear Regression**:

* **Purpose:** Predicts a continuous dependent variable based on one or more independent variables.
* **Output:** A continuous value (e.g., house price, temperature, sales amount).
* **Equation:** Uses a linear equation to model the relationship (Y = a + bX).
* **Error Measurement:** Uses the least squares method to minimize the sum of squared differences between actual and predicted values.
* **Example:** Predicting a student's test score based on hours studied.

**Logistic Regression**:

* **Purpose:** Predicts the probability of a categorical outcome (often binary) based on one or more independent variables.
* **Output:** A probability (a value between 0 and 1) representing the likelihood of an event occurring.
* **Equation:** Uses a logistic function (sigmoid function) to transform the output into probabilities.
* **Error Measurement:** Uses maximum likelihood estimation (MLE) to find the best model parameters.
* **Example:** Predicting whether a customer will click on an ad or not.

1:30 PM TO 3: PM

**DECISION TREES & RANDOM FOREST & SVM**

Decision trees and random forests are supervised machine learning algorithms used for classification and regression, while Support Vector Machines (SVMs) are another supervised learning model primarily used for classification, with each having distinct strengths and weaknesses.

**Decision Trees**:

* **Concept:** Decision trees use a tree-like structure to represent decisions and their possible consequences, branching based on specific conditions or features to classify data.
* **Pros:** Simple to understand and interpret, efficient for data with categorical features, and can handle non-linear data.
* **Cons:** Prone to overfitting, and their accuracy might not be as high as other models like Random Forest or SVM.

**Random Forest**:

* **Concept:** Random forests are an ensemble of decision trees, where multiple trees are trained on different subsets of the data and features to make predictions, and the final prediction is determined by averaging or voting on the predictions of individual trees.
* **Pros:** More robust to overfitting than a single decision tree, handles large datasets and feature spaces effectively, and can achieve high accuracy.
* **Cons:** Can be less interpretable than single decision trees, and computationally intensive for very large datasets or complex trees.

**Support Vector Machines (SVMs):**

* **Concept:** SVMs find the optimal hyperplane that best separates different classes in a feature space, maximizing the margin between the hyperplane and the nearest data points from each class (support vectors).
* **Pros:** Effective in high-dimensional spaces, can handle both linear and non-linear data using kernel tricks, and can be efficient for classification tasks.
* **Cons:** Can be computationally expensive for large datasets, and may not be as interpretable as decision trees.

3:00PM TO 4:30 PM

**NAIVE BAYES CLASSIFICATION & KNN**

Naive Bayes and K-Nearest Neighbors (KNN) are both supervised machine learning algorithms used for classification, but they differ in their approach. Naive Bayes is a probabilistic classifier based on Bayes' theorem, while KNN classifies data points based on the majority class of their nearest neighbors.

Here's a more detailed comparison:

1. **Naive Bayes Classifier**:

* **Principle:** Based on Bayes' theorem, it calculates the probability of a data point belonging to a specific class based on its features.
* **Assumption:** Assumes that the presence of one feature does not influence the presence of another feature (naive assumption).
* **Training:** Requires training data to learn the class probabilities and conditional probabilities of features.
* **Speed:** Generally faster than KNN, especially for large datasets, because it doesn't require distance calculations for each new data point.
* **Complexity:** Relatively simple algorithm with a linear decision boundary.
* **Applications:** Text classification, spam filtering, and document categorization.

2. **K-Nearest Neighbors (KNN) Classifier**:

* **Principle:** Classifies a new data point based on the majority class of its 'k' nearest neighbors in the training data.
* **Training:** Does not involve explicit training; it simply stores the training data.
* **Speed:** Slower than Naive Bayes, especially for large datasets, as it needs to calculate distances between the new data point and all training points.
* **Complexity:** Can model complex, non-linear decision boundaries.
* **Applications:** Image recognition, recommendation systems, and fraud detection.

11.00 PM TO 4.00 AM

**MODEL SELECTION AND MODEL BUILDING**

Model selection and building involve choosing the best statistical model from a set of candidates based on performance criteria, aiming to find a model that accurately fits the data without overfitting. This process includes techniques like variable selection and evaluating model performance.

Here's a more detailed explanation:

**What is Model Selection?**

* Model selection is the process of choosing the most appropriate statistical model from a set of potential models.
* In the context of machine learning and statistical analysis, it involves selecting a model that best fits the data and generalizes well to unseen data.
* The goal is to find a balance between model complexity and predictive accuracy, avoiding overfitting (a model that performs well on the training data but poorly on new data).
* Model selection can also involve designing experiments to collect data that is well-suited for the problem of model selection.

**What is Model Building?**

* Model building is the process of developing a model that accurately describes the relationship between variables.
* It involves identifying relevant variables, determining the form of the relationship (e.g., linear or non-linear), and selecting appropriate model parameters.
* In regression analysis, model building aims to create a model that can predict the value of a dependent variable based on the values of independent variables.
* Model building is an iterative process, where the model is refined and improved based on data analysis and evaluation.

**Key Concepts in Model Selection and Building**:

* **Overfitting:** A model that fits the training data too closely, including noise and irrelevant details, resulting in poor generalization to new data.
* **Underfitting:** A model that is too simple and fails to capture the underlying patterns in the data, resulting in poor performance on both training and new data.
* **Generalization:** The ability of a model to perform well on new, unseen data.
* **Model Complexity:** The number of parameters or variables in a model.
* **Resampling Methods:** Techniques like cross-validation and bootstrapping that are used to estimate the performance of a model on unseen data.
* **Variable Selection:** The process of choosing the most relevant variables to include in a model.
* **Model Evaluation:** Assessing the performance of a model using metrics like accuracy, precision, recall, or F1-score.

**Common Model Selection Techniques**:

* **Train/Test Split:** Splitting the data into training and testing sets to evaluate the model's performance on unseen data.
* **Cross-Validation:** A resampling technique that involves splitting the data into multiple folds and using different combinations of folds for training and testing.
* **Information Criteria (AIC, BIC):** Statistical measures that penalize model complexity, helping to choose the model that best balances fit and complexity.
* **Forward Selection/Backward Elimination:** Stepwise methods for adding or removing variables from a model based on their statistical significance.
* **Regularization:** Techniques like L1 and L2 regularization that penalize complex models, helping to prevent overfitting.

**Factors to Consider When Selecting a Model**:

* **Data Characteristics:** The type of data, the number of observations, and the presence of outliers.
* **Problem Complexity:** The complexity of the problem and the relationships between variables.
* **Model Assumptions:** The assumptions that the model makes about the data and the relationships between variables.
* **Interpretability:** The ease with which the model's results can be understood and explained.
* **Computational Resources:** The computational resources required to train and evaluate the model.

**16/03/25**

12:30 AM TO 4:30 AM

PHASE 1 PROJECT CODE IMPLEMENTATION

**PROBLEM STATEMENT – “ EMOTION DETECTION IN TEXT ”**

**EXPLAINATION OF PROBLEM STATEMENT :**

***Introduction***

Emotion detection in text is a Natural Language Processing (NLP) task that involves classifying text into various emotional categories such as happiness, sadness, anger, surprise, etc. This project utilizes machine learning models to analyze textual data and predict emotions.

***Project Workflow***

The project follows these key steps:

1. **Data Collection**
   * Gather datasets containing text labeled with emotions.
   * Preprocess the data to remove noise.
2. **Data Preprocessing**
   * Tokenization (splitting text into words)
   * Stopword removal
   * Lemmatization/Stemming
   * Handling special characters and punctuation
3. **Model Selection**
   * Choose a machine learning model:
     + Traditional models: Naïve Bayes, SVM, Random Forest
   * Train the model using a labeled dataset.
4. **Model Evaluation**
   * Use evaluation metrics like:
     + Accuracy
     + Precision, Recall, F1-score
     + Confusion matrix
5. **Deployment**
   * Create a web-based interface using HTML, CSS, and JavaScript.
   * Use Flask/Django to integrate the trained model into the frontend.
   * Deploy on a cloud platform (Google Colab).

***Implementation***

**Step 1: Install Required Libraries**

#bash

pip install numpy pandas sklearn neattext seaborn transformers flask.

**Step 2: Load and Preprocess Data**

import pandas as pd

import numpy as np

import seaborn as sns

import neattext.functions as nfx

df = pd.read\_csv("/content/emotion\_dataset\_raw.csv")

df.head()

df['Emotion'].value\_counts()

sns.countplot(x='Emotion',data=df)

df['Clean\_Text'] = df['Text'].apply(nfx.remove\_userhandles

dir(nfx)

df['Clean\_Text'] = df['Clean\_Text'].apply(nfx.remove\_stopwords)

df

**Step 3: Splitting the data into Train and test set Model**

x = df['Clean\_Text']

y = df['Emotion']

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=0.3,random\_state=42)

#Training the Model

from sklearn.pipeline import Pipeline

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.svm import SVC

from sklearn.ensemble import RandomForestClassifier

from sklearn.linear\_model import LogisticRegression

pipe\_lr = Pipeline(steps=[('cv',CountVectorizer()),('lr',LogisticRegression())])

pipe\_lr.fit(x\_train,y\_train)

pipe\_lr.score(x\_test,y\_test)

pipe\_svm = Pipeline(steps=[('cv',CountVectorizer()),('svc', SVC(kernel = 'rbf', C = 10))])

pipe\_svm.fit(x\_train,y\_train)

pipe\_svm.score(x\_test,y\_test)

pipe\_rf = Pipeline(steps=[('cv',CountVectorizer()),('rf', RandomForestClassifier(n\_estimators=10))])

pipe\_rf.fit(x\_train,y\_train)

pipe\_rf.score(x\_test,y\_test)

import joblib

pipeline\_file = open("text\_emotion.pkl","wb")

joblib.dump(pipe\_lr,pipeline\_file)

pipeline\_file.close()

**Step 4: Deploy Model Using Flask**

python

CopyEdit

from flask import Flask, request, jsonify

import pickle

app = Flask(\_\_name\_\_)

# Load trained model and vectorizer

model = pickle.load(open("emotion\_model.pkl", "rb"))

vectorizer = pickle.load(open("vectorizer.pkl", "rb"))

@app.route('/predict', methods=['POST'])

def predict():

text = request.json['text']

text\_vector = vectorizer.transform([text])

prediction = model.predict(text\_vector)[0]

return jsonify({'emotion': prediction})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**4. Evaluation Metrics**

* **Accuracy:** Measures overall correctness.
* **Precision & Recall:** Measures the correctness of each emotion.
* **Confusion Matrix:** Shows misclassifications.

**5. Deployment**

* **Frontend:** HTML, CSS, JavaScript
* **Backend:** Flask/Django
* **Cloud Deployment:** Google Colab / Virtual Studio

**6. Conclusion**

* The developed model provides accurate Emotion Detection In Text.
* Can detect different Emotions.
* Future improvements can include real-time data integration and deep learning models**.**

**7. References**

* Kaggle datasets
* Research on Wikipedia about chatbots And Emotions.

Data Collection and Preparation:

* **Gather Data:** Collect a large dataset of types of Emotions.
* **Data Cleaning:** Handle Unidentical Emotions in the data.
* **Data Transformation:** Convert categorical data (e.g., make, model) into numerical representations suitable for machine learning models.

Machine Learning Model Selection and Training:

* **Regression Models:** Use regression models like linear regression, decision trees, random forests, or gradient boosting machines to Detect Correct Emotion in Text.
* **Model Training:** Train the chosen model on the prepared dataset to Identify the relationship Emotions In Text.

Webpage Development:

* **Frontend:** Create a user-friendly interface where users can give the text as an input.
* **Backend:** Implement the machine learning model and logic to Detect Emotion.
* **API Integration:** Use an API to allow the frontend to communicate with the backend and retrieve Detection.

4. Tools and Technologies:

* **Programming Languages:** python is a popular choice for machine learning, with libraries like Pandas, NumPy, Scikit-learn, and Neattext
* **Web Frameworks:** Flask or Django (Python) can be used to build the backend API.
* **Databases:** Use a database (e.g., PostgreSQL, MySQL) to store and manage the car sales data.
* **Cloud Platforms:**Consider using cloud platforms (e.g., Google colab or virtual studio..etc..,) for hosting the webpage and model deployment.

Example of a Machine Learning Algorithm:

* **Random Forest Regression:** This algorithm combines multiple decision trees to detection Emotion, often leading to better accuracy than a single decision tree.

**In this Hackathon our project was completed till the collection of data set and implementation of backend, and Pushed our code into our team repository.**

**GitHub link :**

**FINAL PROJECT REVIEWS**

**EMOTION DETECTION IN TEXT**

KURUPUDI SIVA DURGA NAGA GOVINDU [ 246Q5A4602 ]

MOLLETI SATYANARAYANA [ 236Q1A4607 ]

GARIKE SRIDEVI [ 236Q1A4604 ]

NOKKI PRIYA DARSHINI [ 236Q1A4602 ]

KUNCHE SURYA SRI VENKATA SAI [ 236Q1A4611]

DODDI DURGA PRASAD [ 236Q1A4609 ]

REVIEW LINK :