PHASE-2 SUBMISSION TEMPLATE

**Student Name:** KAVYASRI R

**Register Number:** 622623205026

**Institution:** Shreenivasa Engineering College

**Department:** B.Tech Information Technology

**Date of Submission:** 08-05-202

**Github Repository Link:**

<https://github.com/Kavyasri2410/Ammu/blob/4f70c0784bd3a6c48f41552855a60cd0f8f9060e>

/KAVYASRI%20R%202.pdf

# )

**TOPIC: Predicting Customer Churn Using Machine Learning to Uncover Hidden Patterns**

# 1. What Is Customer Churn?

Customer churn (also known as customer attrition) refers to the loss of customers over a given period. Businesses aim to predict churn so they can take preventive measures, like targeted offers or improved customer service

**2. Data Required**

Effective churn prediction depends on high-quality data. Typical features include:

Customer demographics (age, gender, location)

Account information (tenure, contract type, billing method)

Usage behavior (logins, product use frequency, session duration)

Customer support interactions (number of tickets, satisfaction scores)

# 3. Machine Learning Pipeline for Churn Prediction

# Step 1: Data Collection

# Collect data from CRM systems, user logs, billing systems, and customer feedback.

# Step 2: Data Preprocessing

# Handle missing values

# Encode categorical features (e.g., One-Hot Encoding, Label Encoding)

# Normalize or scale numerical features

# Feature engineering (e.g., session averages, frequency counts)

# Step 3: Exploratory Data Analysis (EDA)

## Visualize correlations

# Identify feature importance

# Detect class imbalance

# Step 4: Model Selection

# Common models used for churn prediction:

# 3Logistic Regression (baseline)

# Random Forest

# Gradient Boosting (e.g., XGBoost, LightGBM)

# Support Vector Machine

# Neural Networks (for more complex datasets)

# AutoML platforms like Google AutoML or H2O.ai

# Step 5: Model Evaluation

# Use metrics appropriate for classification:

# Accuracy (not always reliable in imbalanced datasets)

# Precision, Recall, F1-Score

# AUC-ROC Curve

# Step 6: Model Deployment

# Integrate the model into a real-time system

# Score customers regularly

# Trigger retention actions based on churn probability

# 5. Uncovering Hidden Patterns

# ML models like decision trees and gradient boosting can reveal:

# Which features are most predictive of churn

# Interaction effects (e.g., tenure \* usage rate)

# Segment-specific risk factors (e.g., high churn among short-term contracts)

# SHAP (SHapley Additive exPlanations) and LIME are useful tools for interpreting complex model decisions and uncovering what drives predictions.

# 6. Business Impact

# By acting on churn predictions, companies can:

# Reduce customer loss

# Improve lifetime value (CLV)

# Personalize retention efforts

# Optimize marketing spend

# 7. Example Use Case

# A telecom company uses historical customer behavior data to train an XGBoost model. The model finds that customers on month-to-month contracts with high support ticket counts and declining usage are most likely to churn. The company offers these users incentives to renew their contracts, reducing churn by 18%.

# 8. Tools & Platforms

# Python/R (pandas, scikit-learn, XGBoost, TensorFlow)

# Cloud ML Platforms: AWS SageMaker, Azure ML, Google Cloud AI

# BI Dashboards: Power BI, Tableau for visualization

# Top of Form

# Bottom of Form

Through natural language explanations and visual elements such as graphs, heatmaps, and timelines, the platform simplifies the complexity of emotion analytics. Alerts and summaries will be provided for sudden emotional shifts or sentiment spikes, enabling proactive decision-making. Additionally, the system will ensure transparency by offering confidence scores for predictions and highlighting which phrases or patterns influenced the emotional classification. Overall, the UX aims to make advanced emotion recognition both accessible and impactful, turning raw social media data into meaningful human insights.

Technical Problems:

### **1. Informal and Noisy Language**

* Social media posts often include slang, emojis, abbreviations, misspellings, and non-standard grammar.
* Example: "I’m ded 😂😂😂" might imply amusement, not death.

### **2. Context Understanding**

* Emotion often depends on context, sarcasm, or previous posts.
* Example: "Great, just failed my test" can sound positive without context but is actually negative.

### 3. **Sarcasm and Irony Detection**

* Sarcasm often inverts the intended sentiment.
* Difficult for models to detect without deep contextual or tonal understanding.

### 4. **Multilingual and Code-Switched Text**

* Social media often involves multiple languages or code-switching within the same post.
* Requires multilingual NLP capabilities.

### 5. **Emojis and Non-Textual Content**

* Emojis carry strong emotional cues, but interpreting them correctly is non-trivial.
* Images, GIFs, and memes also convey emotion but are hard to analyze with text-only methods.

### 6. **Ambiguity in Emotion Categories**

* Emotions are nuanced (e.g., anger vs. frustration vs. disappointment).
* Labeling them accurately requires a fine-grained model and high-quality datasets.

### 7. **Short Texts and Limited Context**

* Tweets and posts are often short, lacking context for accurate sentiment/emotion detection.

### 8. **Bias in Training Data**

* Sentiment models may reflect demographic, political, or cultural biases from training data.
* This leads to inaccurate emotion detection, especially across diverse groups.

### 9. **Real-Time Processing Requirements**

* For real-time monitoring, the system must handle high volume and velocity of data efficiently.

### 10. **Dynamic Language Trends**

* Slang and usage evolve rapidly on social media, requiring continuous model updates.

# 9. Feature Engineering

### **1. Text-Based Features**

* **Bag of Words (BoW)**
  + Converts each word in the text into a feature column.
  + Each column will represent the presence (1) or absence (0) of a word in the text.
* **Term Frequency-Inverse Document Frequency (TF-IDF)**
  + Measures the importance of a word in a document relative to the corpus.
  + Helps in reducing the effect of frequently appearing words like “the” and “is”.
* **N-grams**
  + Capture sequences of words (bigrams, trigrams) to understand context.
  + Example: "not good" → **bigram**: ("not", "good")
* **Sentiment Lexicons**
  + Calculate the presence of sentiment words using a predefined lexicon.
  + E.g., Use **VADER**, **SentiWordNet**, or **AFINN** lexicons for sentiment polarity scoring.

### **2. Text Length Features**

* **Text Length (Number of Words / Characters)**
  + Captures the verbosity of the post, which may correlate with sentiment or emotion intensity.
* **Average Word Length**
  + Average number of characters per word in a post.

### **3. Punctuation & Capitalization Features**

* **Number of Exclamation Marks or Question Marks**
  + The presence of exclamation marks often correlates with strong emotion (excitement, anger).
  + Similarly, question marks may indicate uncertainty or curiosity.
* **Capitalization**
  + Number of capitalized words can signal emphasis or excitement (e.g., "I AM SO HAPPY!!").

### **4. Emoji Features**

* **Emoji Count**
  + Emojis are often strong indicators of emotions (e.g., 😊, 😡). Count the number of emojis in a post.
* **Emoji Sentiment**
  + Map emojis to predefined emotions or sentiment classes and track their occurrences.

### **5. Named Entity Recognition (NER) Features**

* **Entity Counts**
  + Identify and count named entities like people, locations, organizations that appear in the text.
  + Can give insights into topics that may influence sentiment.

### **6. Temporal Features**

* **Time-Based Features**
  + **Hour of Day**, **Day of Week**: Social media posts may show different emotions based on time.
  + Example: People might be more positive in the morning or during weekends.

### **7. Textual Complexity Features**

* **Readability Scores**
  + Use readability formulas like **Flesch-Kincaid** to measure the complexity of the text.

### **8. Hashtags and Mentions**

* **Hashtag Count**
  + The number of hashtags in a post, as hashtags are commonly used to express specific topics or sentiments.
* **Mention Count**
  + Count how many user mentions (@username) appear in the text. This might indicate interactions or influence emotions in social conversations.

# 10. Model Building

### **1. Define the Objective**

* **Classification Task**: Predict either **sentiment** (positive, negative, neutral) or **emotion** (e.g., joy, anger, fear, sadness).
* **Input**: Preprocessed social media text.
* **Output**: Predicted sentiment or emotion label.

### 2**. Choose the Modeling Approach**

Suitable for smaller datasets or explainability-focused applications.

* **Logistic Regression**
* **Naive Bayes**
* **Support Vector Machine (SVM)**
* **Random Forest / Gradient Boosting (e.g., XGBoost)**
* **LSTM / BiLSTM (Recurrent Neural Networks)**
* **CNN for Text Classification**
* **Transformer-based Models (BERT, RoBERTa, DistilBERT)**

### 3**. Model Evaluation**

Use appropriate evaluation metrics:

| **Metric** | **Use Case** |
| --- | --- |
| Accuracy | General performance |
| Precision / Recall | When false positives or false negatives matter |
| F1-Score | Balanced metric for emotion classification |
| Confusion Matrix | Visualize correct vs. incorrect predictions |
| ROC-AUC | For binary or probabilistic sentiment analysis |

### 4**. Hyperparameter Tuning**

* Use **GridSearchCV** or **RandomizedSearchCV** for ML models.
* For deep learning, tune:
  + Learning rate
  + Epochs
  + Batch size
  + Dropout rate

### **5. Final Testing and Deployment**

* Once the model performs well, test it on unseen real-world samples.
* Export the model (.pkl, .pt, or .h5) for deployment via a web app or dashboard.

# 

# 11. Tools and Technologies Used

### **1. Data Collection**

| **Tool/Tech** | **Purpose** |
| --- | --- |
| **Twitter API (Tweepy)** | To collect tweets using keywords, hashtags, or user handles. |
| **Reddit API (PRAW)** | To scrape Reddit posts and comments. |
| **Facebook Graph API** | For public page or group data (limited). |
| **YouTube Data API** | For comment sentiment/emotion mining. |
| **BeautifulSoup / Selenium** | For scraping additional data if APIs are restricted. |

### **2. Data Preprocessing**

| **Tool/Library** | **Purpose** |
| --- | --- |
| **Python** | Main programming language used. |
| **NLTK** | Tokenization, stopword removal, lemmatization. |
| **spaCy** | Named Entity Recognition (NER), dependency parsing. |
| **re** (Regex) | Pattern matching and cleaning text. |
| **langdetect** | Language detection and filtering. |
| **emoji / emot** | Extract or demojize emojis for sentiment clues. |

### **3. Feature Engineering**

| **Tool/Library** | **Purpose** |
| --- | --- |
| **Scikit-learn (sklearn)** | TF-IDF, n-grams, feature selection, ML models. |
| **TextBlob / VADER** | Sentiment scoring using lexicons. |
| **Gensim** | Word2Vec, topic modeling. |
| **Transformers (HuggingFace)** | Pretrained embeddings like BERT, RoBERTa. |

### **4. Model Building**

| **Tool/Library** | **Purpose** |
| --- | --- |
| **Scikit-learn** | Logistic Regression, SVM, Random Forest, Naive Bayes. |
| **TensorFlow / Keras** | Deep learning models like LSTM, CNNs. |
| **PyTorch** | Fine-tune BERT and transformer-based models. |
| **Hugging Face Transformers** | Pretrained models for emotion classification. |

### **5. Visualization**

| **Tool/Library** | **Purpose** |
| --- | --- |
| **Matplotlib** | Basic plotting (bar, line, histogram). |
| **Seaborn** | Heatmaps, distribution plots, confusion matrix. |
| **Plotly / Dash** | Interactive dashboards and visualizations. |
| **WordCloud** | Word cloud generation for emotion-based keywords. |
| **Pandas Profiling / Sweetviz** | Automated EDA reports. |

### **6. Model Deployment (Optional)**

| **Tool/Platform** | **Purpose** |
| --- | --- |
| **Flask / FastAPI** | REST API to serve the model. |
| **Streamlit / Gradio** | Interactive front-end for live sentiment analysis. |
| **Docker** | Containerization of the application. |
| **Heroku / AWS / GCP** | Cloud hosting for real-time applications. |

### 7. Summary

| **Category** | **Technologies** |
| --- | --- |
| **Language** | Python |
| **Libraries** | NLTK, spaCy, scikit-learn, Transformers |
| **Visualization** | Matplotlib, Seaborn, WordCloud, Plotly |
| **Modeling** | Scikit-learn, TensorFlow, PyTorch |
| **Deployment** | Flask, Streamlit, Docker, Heroku |

# 11.Team Members and Contribution

**1. KAVYASRI R– Project Manager & Backend Developer**

* Designs and implements the backend architecture using tools like Firebase or Node.js.

**2. ISWARYA M – Frontend Developer**

* Develops the user interface using technologies such as HTML, CSS, JavaScript, and React.

**3. SANIYA K – UI/UX Designer**

* Designs wireframes and visual mockups for the application..

**4. DIANA S – Data Analyst & API Integrator**

* Collects and prepares datasets such as regional holidays and time zone data.

# 12. Team Members

|  |  |  |
| --- | --- | --- |
| ****S.NO**** | ****NAME**** | ****ROLL NO**** |
| **01** | **Kavyasri R** | **622623205026** |
| **02** | **Iswarya M** | **622623205020** |
| **03** | **Saniya K** | **622623205046** |
| **04** | **Diana S** | **622623205014** |