Project Synopsis Submitted By: Ayushi Dadhich, Marmik Parashar, Shashwat Mishra, Vansh Sachdeva

**Project Title: AI-Driven Customer Churn Prediction for SaaS Companies**

---

**Project Synopsis:**

This project focuses on building a machine learning model to predict customer churn for SaaS companies by analyzing customer usage patterns, subscription behavior, and support interaction data. The model will help businesses proactively identify at-risk customers and develop retention strategies to improve customer loyalty and revenue.

---

**Objective:**

- Develop a predictive model to estimate the likelihood of customer churn.
- Identify key factors contributing to churn and provide actionable insights for retention.
- Build a user-friendly tool/dashboard for churn prediction and analysis.

---

**Data Sources and Features:**

1. **Data Sources**:
    - Customer subscription history (e.g., start date, end date, plan type).
    - Product usage metrics (e.g., login frequency, feature usage, session duration).
    - Customer support data (e.g., number of tickets raised, resolution time).
    - Demographic data (e.g., company size, industry, region).
2. **Features**:
    - Subscription tenure and plan type.
    - Engagement metrics (e.g., usage frequency, inactivity periods).
    - Support interactions (e.g., complaints resolved, unresolved issues).
    - Historical churn status and payment behavior (e.g., late payments).

---

**Risk Factors:**

- **Class Imbalance**: Churned customers are typically a small proportion of the dataset, which may skew model performance.
- **Data Quality**: Missing or incomplete customer records may affect accuracy.
- **Feature Relevance**: Some features may not directly contribute to churn and could introduce noise.

---

**Data Preprocessing:**

1. Handle missing data with imputation techniques (e.g., mean, median for numerical data, mode for categorical).
2. Normalize numerical features (e.g., MinMaxScaler or StandardScaler).
3. Encode categorical variables using one-hot encoding or label encoding.
4. Balance the dataset using techniques like SMOTE (Synthetic Minority Over-sampling Technique).
5. Split data into training, validation, and test sets (e.g., 70-15-15 split).

---

**Model Selection:**

- **Baseline Models**: Logistic Regression, Decision Trees.

- **Advanced Models**: Random Forest, Gradient Boosting (e.g., XGBoost, LightGBM), Neural Networks.
- **Reasoning**: Advanced models can capture complex patterns in customer behavior and feature interactions, crucial for accurate churn prediction.

---

### Exploratory Data Analysis (EDA):

1. Analyze churn rates by customer segment (e.g., plan type, region).
2. Correlation analysis to identify relationships between features and churn.
3. Visualize key trends using histograms, boxplots, and scatterplots.
4. Identify high-risk features like prolonged inactivity or unresolved support issues.

---

### Model Evaluation:

- **Metrics**:
  - Precision, Recall, F1-Score: To balance false positives and false negatives.
  - AUC-ROC Curve: To assess the model's discriminative ability.
  - Log Loss: For probability-based evaluation.
- **Validation**: Use cross-validation techniques to ensure robustness.

---

### Model Deployment:

1. **Platform**: Deploy the model as an API using Flask, FastAPI, or Django.
2. **Interface**: Build a web-based dashboard for business users to input data and view churn predictions.
3. **Monitoring**: Implement logging and model performance tracking to retrain as new data becomes available.

---

Would you like further details on any of these sections?

# Credit Card Fraud Detection Synopsis

## Project Title:

Credit Card Fraud Detection Using Machine Learning

## Project Synopsis:

This project aims to develop a machine learning model to detect credit card fraud by analyzing transaction patterns and identifying anomalies. The model will help financial institutions minimize losses and enhance customer trust by flagging fraudulent transactions in real-time.

## Objective:

- Develop a predictive model for detecting fraudulent transactions.
- Identify key indicators of fraud and generate actionable insights.
- Create a scalable and user-friendly system for continuous fraud monitoring.

## Data Sources and Features:

1. Data Sources:
   - Historical transaction data (e.g., transaction amount, timestamp, location).
   - Customer demographics (e.g., age, account history).
   - Merchant data (e.g., merchant category, location).
2. Features:
   - Transaction attributes (e.g., amount, frequency, time of transaction).
   - Customer behavior patterns (e.g., usual spending locations, typical purchase categories).
   - Merchant risk levels (e.g., high-risk categories, suspicious locations).

## Risk Factors:

- Class Imbalance: Fraudulent transactions represent a small percentage of total transactions.
- Data Quality: Missing or inconsistent transaction details may affect model accuracy.
- Real-time Processing: Ensuring minimal latency for real-time fraud detection.

## Data Preprocessing:

1. Handle missing data using imputation techniques.
2. Normalize transaction amounts and encode categorical variables.
3. Address class imbalance using methods like SMOTE or undersampling.
4. Split data into training, validation, and test sets.

## Model Selection:

- Baseline Models: Logistic Regression, Decision Trees.
- Advanced Models: Random Forest, Gradient Boosting (e.g., XGBoost), Neural Networks.

Reasoning: Advanced models capture complex patterns in transaction data, critical for accurate fraud detection.

## Exploratory Data Analysis (EDA):

1. Analyze transaction patterns by time, location, and merchant type.
2. Identify correlations between features and fraudulent activity.
3. Visualize key trends using histograms, scatterplots, and time-series analysis.

## Model Evaluation:

- Metrics: Precision, Recall, F1-Score, AUC-ROC, and Log Loss.
- Validation: Cross-validation techniques to ensure robustness.

## Model Deployment:

1. Platform: Deploy as an API using Flask or FastAPI.
2. Interface: Provide a dashboard for real-time transaction monitoring and alerts.
3. Monitoring: Continuous model performance tracking and retraining as new data becomes available.

Project Synopsis Submitted By: Ayushi Dadhich, Marmik Parashar, Shashwat Mishra, Vansh Sachdeva

# Stock Market Trend and Sentiment Analysis Synopsis

## Project Title:

Stock Market Trend and Sentiment Analysis Using Machine Learning and NLP

## Project Synopsis:

This project focuses on leveraging machine learning and natural language processing (NLP) techniques to analyze stock market trends and sentiment from financial news, social media, and historical stock data. The goal is to provide investors and analysts with actionable insights for making informed trading decisions.

## Objective:

- Develop a model to predict stock market trends based on historical data.
- Perform sentiment analysis on financial news and social media to gauge market sentiment.
- Integrate trend and sentiment data to enhance investment decision-making.

## Data Sources and Features:

1. Data Sources:
   - Historical stock price data (e.g., open, close, high, low, volume).
   - Financial news articles and social media posts (e.g., Twitter, Reddit).
   - Economic indicators (e.g., GDP, unemployment rates).
2. Features:
   - Stock price trends (e.g., moving averages, volatility).
   - Sentiment scores from text data (e.g., positive, negative, neutral).
   - Market-related keywords and topics from textual data.

## Risk Factors:

- Data Quality: Inconsistent or noisy textual data from various sources.
- Market Volatility: Sudden market changes can impact prediction accuracy.
- Sentiment Misinterpretation: Sarcasm or complex language can lead to incorrect sentiment analysis.

## Data Preprocessing:

1. Clean and normalize stock price data.
2. Preprocess text data (e.g., tokenization, stopword removal, stemming/lemmatization).
3. Use sentiment lexicons or pre-trained models for sentiment scoring.
4. Encode categorical variables and handle missing values.

Project Synopsis Submitted By: Ayushi Dadhich, Marmik Parashar, Shashwat Mishra, Vansh Sachdeva

## Model Selection:

- For Trend Prediction: ARIMA, LSTM, Random Forest.
- For Sentiment Analysis: Naïve Bayes, SVM, Transformer-based models (e.g., BERT).

Reasoning: Advanced models can capture both sequential patterns in stock data and complex relationships in textual data for accurate predictions.

## Exploratory Data Analysis (EDA):

1. Analyze historical stock price trends and patterns.
2. Perform keyword frequency analysis and topic modeling on textual data.
3. Visualize correlations between sentiment scores and stock price movements.

## Model Evaluation:

- Metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) for trend prediction; Precision, Recall, F1-Score for sentiment analysis.
- Validation: Cross-validation and backtesting to ensure robustness.

## Model Deployment:

1. Platform: Deploy as a web application using Flask, FastAPI, or Django.
2. Interface: Provide interactive dashboards for trend visualization and sentiment insights.
3. Monitoring: Implement continuous monitoring for model performance and retraining as new data is ingested.

Project Synopsis Submitted By: Ayushi Dadhich, Marmik Parashar, Shashwat Mishra, Vansh Sachdeva

# Energy Consumption Trend Analysis Synopsis

## Project Title:

Energy Consumption Trend Analysis Using Machine Learning and Statistical Methods

## Project Synopsis:

This project aims to analyze energy consumption trends across various sectors using machine learning and statistical techniques. By identifying consumption patterns, the model will help policymakers, utility providers, and businesses make informed decisions to optimize energy usage and reduce costs.

## Objective:

- Analyze historical energy consumption data to identify trends and patterns.
- Develop predictive models to forecast future energy demands.
- Provide actionable insights to optimize energy usage and improve efficiency.

## Data Sources and Features:

1. Data Sources:
   - Historical energy consumption records from utility providers.
   - Weather data (e.g., temperature, humidity).
   - Economic and demographic data (e.g., GDP, population).
2. Features:
   - Time-based features (e.g., hour, day, month, season).
   - Weather-related variables (e.g., temperature, rainfall).
   - Sector-specific consumption patterns (e.g., residential, industrial, commercial).

## Risk Factors:

- Data Inconsistency: Incomplete or inconsistent data across different regions or sectors.
- Seasonal Variability: Sudden changes in weather can affect consumption patterns.
- External Influences: Economic and policy changes can impact energy consumption trends.

## Data Preprocessing:

1. Handle missing and inconsistent data using imputation techniques.
2. Normalize and scale numerical data for consistency.
3. Create time-series features and perform lag analysis.
4. Encode categorical variables (e.g., region, sector).

Project Synopsis Submitted By: Ayushi Dadhich, Marmik Parashar, Shashwat Mishra, Vansh Sachdeva

## Model Selection:

- Baseline Models: Linear Regression, Decision Trees.
- Advanced Models: ARIMA, LSTM, Random Forest, Gradient Boosting.

Reasoning: Advanced models capture complex temporal patterns in energy consumption data, essential for accurate forecasting.

## Exploratory Data Analysis (EDA):

1. Analyze energy consumption patterns by sector and region.
2. Perform correlation analysis between weather variables and energy usage.
3. Visualize time-series data using line charts, heatmaps, and seasonal decomposition plots.

## Model Evaluation:

- Metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE).
- Validation: Use time-series cross-validation and backtesting to ensure robustness.

## Model Deployment:

1. Platform: Deploy as a web-based dashboard using Flask or FastAPI.
2. Interface: Provide interactive visualizations and forecasting tools.
3. Monitoring: Implement real-time data ingestion and model performance tracking.