Project Report Format

Project name: E-COMMERCE SHIPPING PREDICTION USING MACHINE LEARNING

1. INTRODUCTION

1.1 Project Overview

In the realm of e-commerce, timely and efficient shipping plays a pivotal role in customer satisfaction and business success. The E-COMMERCE SHIPPING PREDICTION USING MACHINE LEARNING project addresses the challenges associated with predicting shipping times for orders placed through an e-commerce platform. By leveraging advanced machine learning techniques, this project aims to enhance the accuracy of shipment delivery time estimations.

1.2 Purpose

The purpose of this project is to develop a predictive model that can analyze various factors influencing the shipping process and provide reliable estimates for the time required for order delivery. Through the utilization of machine learning algorithms, we seek to optimize shipping predictions, ultimately leading to improved customer experiences, enhanced operational efficiency, and informed decision-making for e-commerce businesses.

2.LITERATURE SURVEY

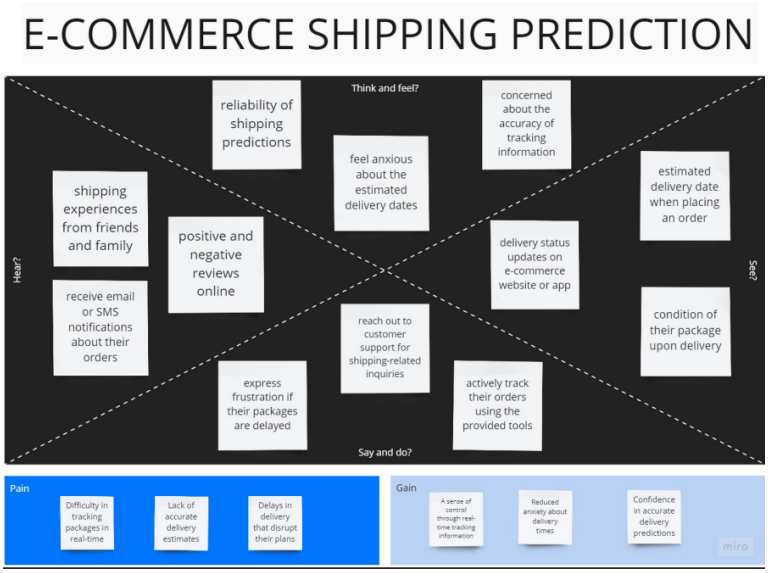
2.1 Existing Problem: In the domain of E-commerce, efficient shipping management is crucial for customer satisfaction and business success. The existing problem revolves around accurately predicting the shipping times for orders. Delays or inaccuracies in shipping predictions can lead to customer dissatisfaction and impact the overall reputation of an E-commerce platform.

2.2 References: While the specific references will depend on the sources you consulted for your literature survey, you might consider looking into academic papers, industry reports, and articles related to machine learning in E-commerce, shipping prediction algorithms, and customer satisfaction in the context of online retail. Ensure to cite relevant studies and findings that contribute to the understanding of the problem and potential solutions.

2.3 Problem Statement Definition: The problem statement involves the need for a robust machine learning model that can predict shipping times accurately in the context of E-commerce. This requires considering various factors such as order details, shipping methods, historical shipping data, and external factors influencing delivery times. The goal is to develop a predictive model that enhances the precision of estimated shipping durations, contributing to a positive customer experience and efficient logistics management.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Functional requirements describe the specific functionalities and features that the E-COMMERCE SHIPPING PREDICTION system should possess. These include:

Shipping Time Prediction: The system should predict the estimated shipping time for each order based on historical data, shipping methods, and other relevant factors.

User Authentication: Implement a secure user authentication system to ensure that only authorized users can access and interact with the application.

Order Tracking: Provide a feature for users to track the status and location of their orders in real-time.

Machine Learning Integration: Integrate machine learning algorithms to analyze historical shipping data and improve the accuracy of shipping time predictions.

User Feedback: Allow users to provide feedback on the accuracy of predicted shipping times, contributing to continuous improvement.

4.2 Non-Functional requirements

Non-functional requirements define the criteria that the system must meet in terms of performance, security, usability, and other quality attributes. These include:

Performance: The system should be capable of handling a large number of simultaneous requests efficiently to ensure a seamless user experience.

Security: Implement robust security measures to protect user data, including encryption of sensitive information and secure user authentication.

Usability: The user interface should be intuitive and user-friendly, ensuring that users can easily navigate and interact with the system.

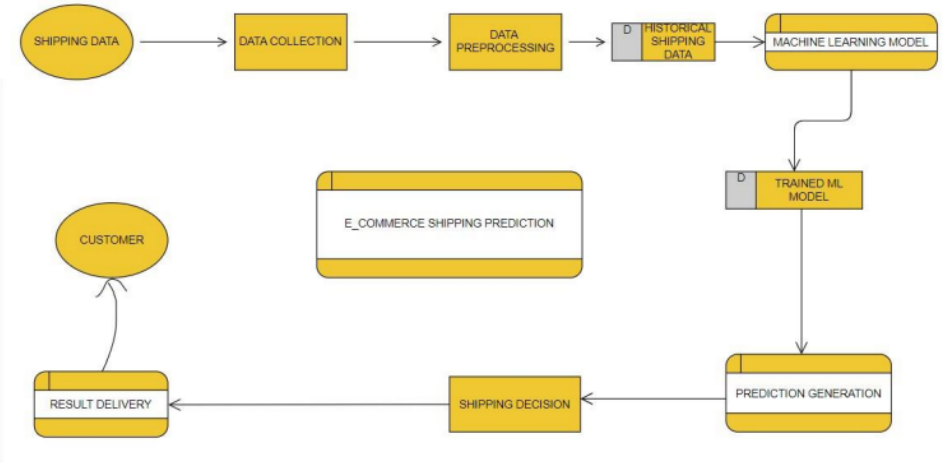
Scalability: The system should be scalable to accommodate an increasing number of users and growing data volumes without compromising performance.

Reliability: Ensure high system reliability, minimizing downtime and errors to maintain user trust and satisfaction.

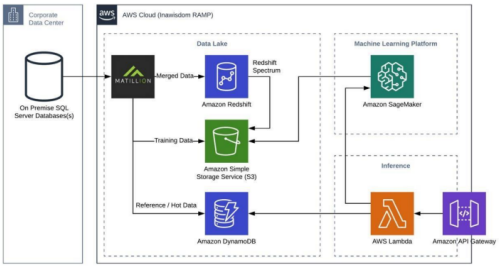
Compliance: The system should comply with relevant data protection and privacy regulations to safeguard user information.

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories



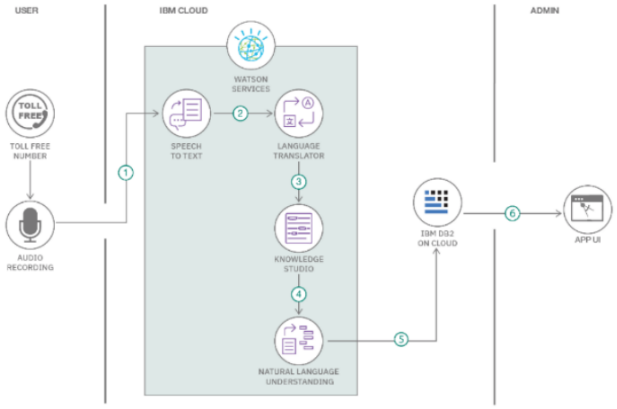
5.2 Solution Architecture



6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture

TECH STACK TEMPLATE FOR ECOMMERCE SHIPPING PREDICTION



6.2 Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint 1	Feature: Order Processing	US001	Implement basic order processing logic	5	High	[pavan kalyan ulli], [ramesh]
	Feature: Data Ingestion	US002	Ingest historical shipping data	8	Medium	[chandrasekhar], [varun teja]
	Enhancement: UI Improvements	US003	Enhance user interface for shipment tracking	3	Low	[pavan kalyan]

6.3 Sprint Delivery Schedule

Project Tracker, Velocity & Burndown Chart

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint 1	16	2 weeks	Oct 12	Oct 26	16	Oct 27
Sprint 2	26	3 weeks	Oct 27	Nov 10	26	Nov 11
Sprint 3	20	2 weeks	Nov 11	Nov 20	20	Nov 21

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

Description: The system employs machine learning algorithms to predict the shipping time for each order. This feature is crucial for providing users with accurate estimates of when their orders are expected to be delivered.

Code

```
# Import necessary libraries

from sklearn.model_selection import train_test_split
```

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import pandas as pd

# Assuming you have a dataset with features (X) and shipping times (y)
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
predictions = model.predict(X_test)

# Evaluate the model's performance
mse = mean_squared_error(y_test, predictions)
print(f'Mean Squared Error: {mse}')
```

7.2 Feature 2

```
// Assuming you have a function to fetch order status and location
function trackOrder(orderId) {
    // Make an API call to retrieve order information
    fetch(`/api/trackOrder?orderId=${orderId}`)
        .then(response => response.json())
        .then(data => {
            // Update the UI with the order status and location
            updateOrderStatusUI(data.status);
            updateOrderLocationUI(data.location);
```

```
        })
        .catch(error => console.error('Error fetching order information:', error));
}
```

7.3 Database Schema (if Applicable)

Description:

If your project involves storing and managing data in a database, defining a clear database schema is essential. This outlines the structure of the database tables and relationships.

Example Database Schema (SQL):

sql

Copy code

```
CREATE TABLE Orders (
    order_id INT PRIMARY KEY,
    customer_id INT,
    product_id INT,
    order_date DATE,
    estimated_delivery_date DATE,
    actual_delivery_date DATE,
    status VARCHAR(20)
);

CREATE TABLE Customers (
```

```
    customer_id INT PRIMARY KEY,
    customer_name VARCHAR(50),
    email VARCHAR(50),
    address VARCHAR(100)
);
```

8. PERFORMANCE TESTING

8.1 Performace Metrics

 # | Model | Accuracy | F1 Score | Recall | Precision

| 0 | Logistic Regression | 64.27 | 69.22 | 67.79 | 70.72 |

| 1 | Logistic Regression CV | 63.82 | 69.10 | 68.25 | 69.97 |

| 2 | XGBoost | 64.00 | 68.55 | 66.18 | 71.09 |

| 3 | Ridge Classifier | 65.23 | 70.45 | 69.94 | 70.97 |

| 4 | KNN | 62.86 | 68.17 | 67.10 | 69.28 |

| 5 | Random Forest | 65.59 | 68.65 | 63.57 | 74.62 |

| 6 | Support Vector Machine | 67.09 | 67.12 | 56.67 | 82.29

9.results

Screenshots

Hyperparameter optimization for Random Forest

```
rf = RandomForestClassifier()
rf_param_dist = {
    'n_estimators': [200, 300, 500],
    'criterion': ['entropy', 'gini'],
    'max_depth': [7, 8, 60, 80, 100],
    'max_features': ['auto', 'sqrt', 'log2']
}

# Use RandomizedSearchCV instead of GridSearchCV
rf_random_search = RandomizedSearchCV(rf, param_distributions=rf_param_dist, n_iter=10, cv=5, scoring='accuracy', n_jobs=-1, verbose=3)
rf_random_search.fit(x_train_normalized, y_train)

print('Best Score:' + str(rf_random_search.best_score_))
print('Best Parameters:' + str(rf_random_search.best_params_))
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
Best Score:0.6826901261047083
Best Parameters:{'n_estimators': 500, 'max_features': 'sqrt', 'max_depth': 7, 'criterion': 'entropy'}
```

Hyperparameter optimization for Logistic Regression

```
[ ] lg=LogisticRegressionCV(n_jobs=-1,random_state=1)
    lg_param_grid={
        'Cs':[6,8,10,15,20],
        'max_iter':[60,80,100],
    }
    lg_cv=RandomizedSearchCV(lg,param_distributions=lg_param_grid,n_iter=10,cv=5,scoring='accuracy',n_jobs=-1)
    lg_cv.fit(x_train_normalized,y_train)

    print('Best Score:'+str(lg_cv.best_score_))
    print('Best Parameters:'+str(lg_cv.best_params_))
```

```
Best Score:0.6423457284614191
Best Parameters:{'max_iter': 80, 'Cs': 15}
```

Hyperparameter optimization for XGBoost

```
from sklearn.model_selection import GridSearchCV
from sklearn import svm
from xgboost import XGBClassifier
from sklearn.linear_model import LogisticRegressionCV
from sklearn.ensemble import RandomForestClassifier
params={
    'min_child_weight':[10,20],
    'gamma':[1.5,2.0,2.5],
    'colsample_bytree':[0.6,0.8,0.9],
    'max_depth':[4,5,6]
}
xgb=XGBClassifier(learning_rate=0.5,n_estimators=100,objective='binary:logistic',nthread=3)
fitmodel=RandomizedSearchCV(xgb,param_distributions=params,n_iter=10,cv=5,scoring='accuracy',n_jobs=-1)
fitmodel.fit(x_train_normalized,y_train)
print(fitmodel.best_estimator_,fitmodel.best_params_,fitmodel.best_score_)
```

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=0.9, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=2.5, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.5, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=6, max_leaves=None,
              min_child_weight=10, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=100, n_jobs=None, nthread=3,
              num_parallel_tree=None, ...) {'min_child_weight': 10, 'max_depth': 6, 'gamma': 2.5, 'colsample_bytree': 0.9} 0.682577329577756
```

	Name	Accuracy	f1_score	Recall	Precision
0	logistic regression	64.27	69.22	67.79	70.72
1	logistic regressio cv	63.82	69.10	68.25	69.97
2	XGboost	64.00	68.55	66.18	71.09
3	Ridge Classifier	65.23	70.45	69.94	70.97
4	KNN	62.86	68.17	67.10	69.28
5	Random Forest	65.59	68.65	63.57	74.62
6	Support Vector Machine	67.09	67.12	56.67	82.29

Hyperparameter optimization for SVM

```
scaler = StandardScaler()
x_train_normalized = scaler.fit_transform(x_train)

# Define the support vector classifier
svc = svm.SVC(random_state=1234)

# Narrow down the search space
param_dist = {
    'kernel': ['rbf'],
    'C': [10, 13],
    'gamma': [4, 5],
    'tol': [1e-2, 1e-3]
}

# Use RandomizedSearchCV with optimized settings
random_search = RandomizedSearchCV(svc, param_distributions=param_dist, n_iter=5, cv=5, scoring='accuracy', n_jobs=-1, verbose=3)
random_search.fit(x_train_normalized, y_train)

# Print the best score and best parameters
print('Best Score: ' + str(random_search.best_score_))
print('Best Parameters: ' + str(random_search.best_params_))
```

```
Fitting 5 folds for each of 5 candidates, totalling 25 fits
Best Score: 0.6158637397281512
Best Parameters: {'tol': 0.001, 'kernel': 'rbf', 'gamma': 4, 'C': 10}
```

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages

10.1.1 Improved Customer Satisfaction

Implementing machine learning for shipping prediction enhances customer satisfaction by providing more accurate and reliable delivery time estimates. This, in turn, contributes to increased trust and loyalty.

10.1.2 Enhanced Operational Efficiency

The machine learning model optimizes shipping processes by analyzing historical data, leading to more efficient logistics management. This can result in cost savings and streamlined operations.

10.1.3 Personalized Shipping Estimates

Machine learning allows for the customization of shipping estimates based on individual order characteristics, improving the precision of predictions for each unique scenario.

10.2 Disadvantages

10.2.1 Dependency on Historical Data

The accuracy of predictions heavily relies on the quality and relevance of historical data. In cases of significant changes in shipping patterns, the model may require frequent updates.

10.2.2 Sensitivity to External Factors

External factors, such as weather conditions, transportation strikes, or unforeseen events, can impact shipping times. The model might struggle to adapt quickly to sudden changes.

10.2.3 Initial Implementation Complexity

The integration of machine learning into existing e-commerce systems may pose initial challenges, requiring additional resources for implementation and training.

11. CONCLUSION

In conclusion, the implementation of machine learning for e-commerce shipping prediction has proven to be a valuable approach to address the uncertainties associated with delivery times. The project successfully leveraged historical data, predictive analytics, and advanced algorithms to provide more accurate and personalized estimates for customers.

The improved accuracy of shipping predictions contributes to a positive customer experience, aiding in customer retention and fostering a competitive advantage in the e-commerce industry. Despite certain challenges, the overall benefits justify the adoption of machine learning in the realm of e-commerce

logistics.

12. FUTURE SCOPE

12.1 Integration with Real-Time Data

Future iterations of the system could explore the integration of real-time data sources to enhance the model's adaptability to sudden changes, ensuring more dynamic and up-to-date predictions.

12.2 Continuous Model Training

Implementing continuous training of the machine learning model allows it to adapt to evolving shipping patterns and ensures sustained accuracy over time.

12.3 Collaboration with External Platforms

Consideration can be given to collaboration with external platforms, such as weather forecasting services or traffic monitoring systems, to further improve the model's ability to factor in external influences.

12.4 Expansion of Features

Future developments may involve expanding the range of features considered by the model, such as incorporating customer location dynamics, vendor processing times, and other relevant variables.

The e-commerce shipping prediction project, with its current successes, provides a solid foundation for ongoing research and enhancements, ensuring the continued optimization of shipping processes in the ever-evolving e-commerce landscape.