Experiment no. 09

Aim : Build model on following data sets in various domains.
a. Machine Learning data set: e.g. Credit Card Fraud Detection Dataset
b. NLP data sets: eg. Twitter Dataset, HotspotQA Dataset

Course Outcome: Analyze different forms of data with respect to different phases of Machine Learning.

Theory :

1. Logistic Regression (LR) :

1. Definition: A statistical model used for binary or multiclass classification, predicting the probability of an outcome.

2. Type: Supervised learning, linear model.

3. Purpose: Predict the likelihood of a class (e.g., fraud or not fraud).

4. Algorithm: Uses the sigmoid function to map predictions to probabilities between 0 and 1.

5. Input: Continuous or categorical features (after encoding).

6. Output: Probability of class membership; can be converted to 0/1 using a threshold (default 0.5).

7. Assumption: Assumes a linear relationship between input features and log-odds of the outcome.

8. Advantages: Simple, interpretable, fast, works well with small datasets.

9. Disadvantages: Struggles with non-linear relationships and complex interactions between features.

10. Example: Predict whether a credit card transaction is fraudulent (Class=0 or Class=1) based on transaction features.

2. Random Forest (RF) :

1. Definition: An ensemble learning method using multiple decision trees to improve prediction accuracy.

2. Type: Supervised learning, classification or regression.

3. Purpose: Combines predictions of multiple trees to reduce overfitting and increase robustness.

4. Algorithm: Trains many decision trees on random subsets of data and features (bagging).

5. Input: Can handle numeric and categorical data.

6. Output: Majority vote of trees for classification; average for regression.

7. Assumption: No strict assumption; robust to non-linear relationships and feature correlations.

8. Advantages: High accuracy, handles large datasets, reduces overfitting.

9. Disadvantages: Can be computationally expensive, less interpretable than a single tree.

10. Example: Classify transactions as fraud/not fraud using ensemble decision trees trained on transaction features.

3. XGBoost (Extreme Gradient Boosting) :

1. Definition: An advanced gradient boosting algorithm for supervised learning that builds sequential trees.

2. Type: Supervised learning, used for classification and regression.

3. Purpose: Improves model accuracy by minimizing errors from previous trees using gradient descent.

4. Algorithm: Builds trees sequentially, each correcting mistakes of the previous one.

5. Input: Numeric features; categorical features must be encoded.

6. Output: Probability (classification) or predicted value (regression).

7. Assumption: Works well with non-linear and complex patterns; no linearity assumption.

8. Advantages: Very high accuracy, handles missing values, regularization reduces overfitting.

9. Disadvantages: Complex, slower to train than single models, needs careful hyperparameter tuning.

10. Example: Predict fraud probability for credit card transactions using gradient boosting on transaction features.

Program :

Credit Card Fraud Detection :

```
import pandas as pd from sklearn.model_selection import
train_test_split from sklearn.preprocessing import
StandardScaler from sklearn.ensemble import
RandomForestClassifier from sklearn.linear_model import
LogisticRegression from xgboost import XGBClassifier from
sklearn.metrics import classification_report, roc_auc_score
from imblearn.over_sampling import SMOTE

# Load dataset
df = pd.read_csv("creditcard.csv")

# Preprocessing
X = df.drop(columns=['Class'])
y = df['Class']

# Scale features scaler
=StandardScaler()
X_scaled = scaler.fit_transform(X)

# Handle imbalance using SMOTE smote
= SMOTE(random_state=42)
X_res, y_res = smote.fit_resample(X_scaled, y)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_res, y_res, test_size=0.2,
random_state=42)

# Model training and evaluation models
= {
   "Logistic Regression": LogisticRegression(max_iter=1000),
   "Random Forest": RandomForestClassifier(),
   "XGBoost": XGBClassifier(use_label_encoder=False, eval_metric='logloss') }

for name, model in models.items():
model.fit(X_train, y_train)     y_pred =
model.predict(X_test)     print(f"{name}
Evaluation:")
print(classification_report(y_test, y_pred))
   print(f"ROC-AUC: {roc_auc_score(y_test, y_pred)}\n")
```

```
# Deployment (CLI prediction using trained Random Forest) final_model
= RandomForestClassifier()
final_model.fit(X_train, y_train)

def predict_fraud(transaction):
    transaction_scaled = scaler.transform([transaction])
prediction = final_model.predict(transaction_scaled)     return
"Fraudulent" if prediction[0] == 1 else "Genuine"

# Example usage
transaction = [0.0, 1.0, -1.0, 0.5, 0.0, -0.3, 0.1, 0.2, -0.1, 0.0, 0.3, -0.2, 0.0, 0.1, -0.1,
0.2, 0.0, -0.2, 0.1, 0.0, -0.3, 0.2, 0.1, -0.1, 0.0, 0.3, -0.2, 0.1, 0.0, -0.1]
print("Prediction:", predict_fraud(transaction))
```

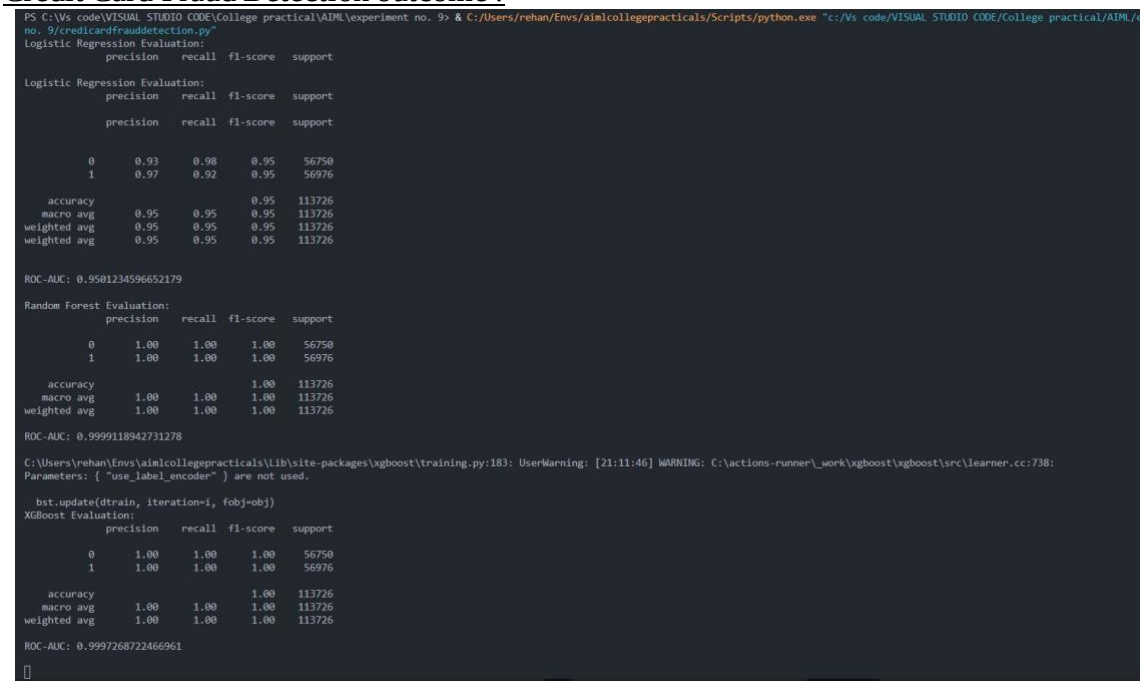
NLP Data Sets (Twitter Data Set) :

```
import pandas as pd
from sklearn.model_selection import train_test_split from
sklearn.feature_extraction.text import TfidfVectorizer from
sklearn.linear_model import LogisticRegression from
sklearn.metrics import classification_report

# Load Twitter dataset (CSV with 'text' and 'label')
df = pd.read_csv("Twitter_Data.csv")  # Make sure this CSV exists

# Preprocessing: lowercase
df['text'] = df['text'].str.lower()

# Split data
X_train, X_test, y_train, y_test = train_test_split(df['text'], df['label'], test_size=0.2,
random_state=42)

# Vectorize text
vectorizer = TfidfVectorizer(stop_words='english', max_features=5000)
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)

# Train Logistic Regression
model = LogisticRegression(max_iter=1000)
model.fit(X_train_vec, y_train)

# Predict and evaluate y_pred =
model.predict(X_test_vec)
print(classification_report(y_test, y_pred))

# Test a new tweet new_tweet = ["I love
this new phone!"] new_vec =
```

vectorizer.transform(new_tweet) prediction
= model.predict(new_vec)
print("Predicted sentiment:", prediction[0])

Output :

Credit Card Fraud Detection outcome :



```
PS C:\Vs code\VISUAL STUDIO CODE\College practical\AIML\experiment no. 9> & C:/Users/rehan/Envs/aimlcollegepracticals/Scripts/python.exe "c:/Vs code/VISUAL STUDIO CODE/College practical/AIML/e
no. 9/credicardfrauddetection.py"
Logistic Regression Evaluation:
              precision    recall  f1-score   support

Logistic Regression Evaluation:
              precision    recall  f1-score   support

              precision    recall  f1-score   support


           0       0.93      0.98      0.95     56750
           1       0.97      0.92      0.95     56976

    accuracy                           0.95    113726
   macro avg       0.95      0.95      0.95    113726
weighted avg       0.95      0.95      0.95    113726
weighted avg       0.95      0.95      0.95    113726


ROC-AUC: 0.9501234596652179

Random Forest Evaluation:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56750
           1       1.00      1.00      1.00     56976

    accuracy                           1.00    113726
   macro avg       1.00      1.00      1.00    113726
weighted avg       1.00      1.00      1.00    113726

ROC-AUC: 0.9999118942731278

C:\Users\rehan\Envs\aimlcollegepracticals\Lib\site-packages\xgboost\training.py:183: UserWarning: [21:11:46] WARNING: C:\actions-runner\_work\xgboost\xgboost\src\learner.cc:738:
Parameters: { "use_label_encoder" } are not used.

  bst.update(dtrain, iteration=i, fobj=obj)
XGBoost Evaluation:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56750
           1       1.00      1.00      1.00     56976

    accuracy                           1.00    113726
   macro avg       1.00      1.00      1.00    113726
weighted avg       1.00      1.00      1.00    113726

ROC-AUC: 0.9997268722466961
```

NLP data set outcome :

```
PS C:\Vs code\VISUAL STUDIO CODE\College practical\AIML\experiment no. 9> & C:/Users/rehan/Envs/aimlcollegepracticals/Scripts/python.exe "c:/Vs code/VISUAL STUDIO CODE/College practical/AIML
/experiment no. 9/twitter.py"
Classification Report:

              precision    recall  f1-score   support

        -1.0       0.87      0.75      0.80      7230
         0.0       0.80      0.96      0.87     10961
         1.0       0.90      0.83      0.86     14404

    accuracy                           0.85     32595
   macro avg       0.86      0.84      0.84     32595
weighted avg       0.86      0.85      0.85     32595

Tweet: I love this new phone!
Predicted category: 1.0

Tweet: Modi government did a terrible job!
Predicted category: -1.0

Tweet: Looking forward to the upcoming elections.
Predicted category: 0.0

PS C:\Vs code\VISUAL STUDIO CODE\College practical\AIML\experiment no. 9>
```

Conclusion :

I have successfully created the models for Credit Card Detection program and used the Twitter Dataset