Assignment # 8

[Find errors and complete the steps]

[Note]

Use the PyCharm software if google colab is not working properly

Sales Forecasting for a Retail Store

```
# Sales Forecasting for a Retail Store

data_sales = pd.read_csv("retail_sales.csv")  # Ensure dataset contains
relevant features

X_sales = data_sales[['ad_budget', 'discount_rate', 'season',
'store_traffic']]
y_sales = data_sales['sales']

X_train_sales, X_test_sales, y_train_sales, y_test_sales =
train_test_split(X_sales, y_sales, test_size=0.2, random_state=42)

model_sales = LinearRegression()
model_sales.fit(X_train_sales, y_train_sales)

y_pred_sales = model_sales.predict(X_test_sales)

mse_sales = mean_squared_error(y_test_sales, y_pred_sales)
r2_sales = r2_score(y_test_sales, y_pred_sales)

print(f"Sales Forecast - MSE: {mse_sales}")
print(f"Sales Forecast - R-squared: {r2_sales}")

plt.scatter(y_test_sales, y_pred_sales)
```

```
plt.xlabel("Actual Sales")
plt.ylabel("Predicted Sales")
plt.title("Actual vs Predicted Sales")
plt.show()
```

[Your Task]

1. Get retail_sales.csv from Kaggle dataset or google
2. Install the dependencies
3. Run the above code
4. Resolve the error
5. Run the code again
6. Show the output

Email Spam Detection using SVM

Classifies emails as spam or ham (not spam) using a text classification approach with SVM.

```
import pandas as pd
import numpy as np
import re
import nltk
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report

# Load dataset
df = pd.read_csv("https://raw.githubusercontent.com/justmarkham/pycon-2016-tutorial/master/data/sms-spam-collection.csv", encoding='latin-1')
df.columns = ["label", "message"]

# Convert labels to binary (ham = 0, spam = 1)
```

```
df['label'] = df['label'].map({'ham': 0, 'spam': 1})

# Text Preprocessing
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer

ps = PorterStemmer()
corpus = []
for msg in df['message']:
    msg = re.sub('[^a-zA-Z]', ' ', msg).lower().split()
    msg = [ps.stem(word) for word in msg if word not in
stopwords.words('english')]
    corpus.append(" ".join(msg))

# Convert text into numerical features using TF-IDF
vectorizer = TfidfVectorizer(max_features=5000)
X = vectorizer.fit_transform(corpus).toarray()
y = df['label'].values

# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Train SVM Model
svm_model = SVC(kernel='linear')
svm_model.fit(X_train, y_train)

# Predict and Evaluate
y_pred = svm_model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

[Your Task]

1. Get `sms-spam-collection.csv` from Kaggle dataset or google
2. Install the dependencies
3. Run the above code
4. Resolve the error

5. Run the code again
6. Show the output

Customer Churn Prediction using SVM

Predicts whether a customer will churn (leave a subscription-based service) based on customer data.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report

# Load dataset
df = pd.read_csv("https://raw.githubusercontent.com/stedy/Machine-Learning-with-R-datasets/master/churn.csv")

# Drop unnecessary columns
df = df.drop(columns=['customerID'])

# Encode categorical variables
label_enc = LabelEncoder()
for col in df.select_dtypes(include=['object']).columns:
    df[col] = label_enc.fit_transform(df[col])

# Define features and target variable
X = df.drop(columns=['Churn'])
y = df['Churn']

# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Scale features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train SVM Model
svm_model = SVC(kernel='rbf', C=1, gamma='scale')
```

```
svm_model.fit(X_train, y_train)

# Predict and Evaluate
y_pred = svm_model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

[Your Task]

1. Get churn.csv from Kaggle dataset or google
2. Install the dependencies
3. Run the above code
4. Resolve the error
5. Run the code again
6. Show the output

Fraud Detection in Credit Card Transactions

Detects fraudulent credit card transactions using SVM on a Kaggle dataset.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score

# Load dataset
df =
pd.read_csv("https://storage.googleapis.com/download.tensorflow.org/data/cre
ditcard.csv")

# Reduce dataset size for faster processing
df = df.sample(frac=0.1, random_state=42)
```

```
# Define features and target
X = df.drop(columns=['Class'])
y = df['Class']

# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Scale features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train SVM Model
svm_model = SVC(kernel='rbf', C=1, gamma='scale')
svm_model.fit(X_train, y_train)

# Predict and Evaluate
y_pred = svm_model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

[Your Task]

```
1. Install the dependencies
2. Compile and run the code
3. Print and show the output
```