MACHINE LEARNING

LAB 6

Nikki Evana Blessy.N

2341459

Implement KNN regressor to predict the

Plasma glucose concentration. Select the top principal component using PCA

INTRODUCTION:

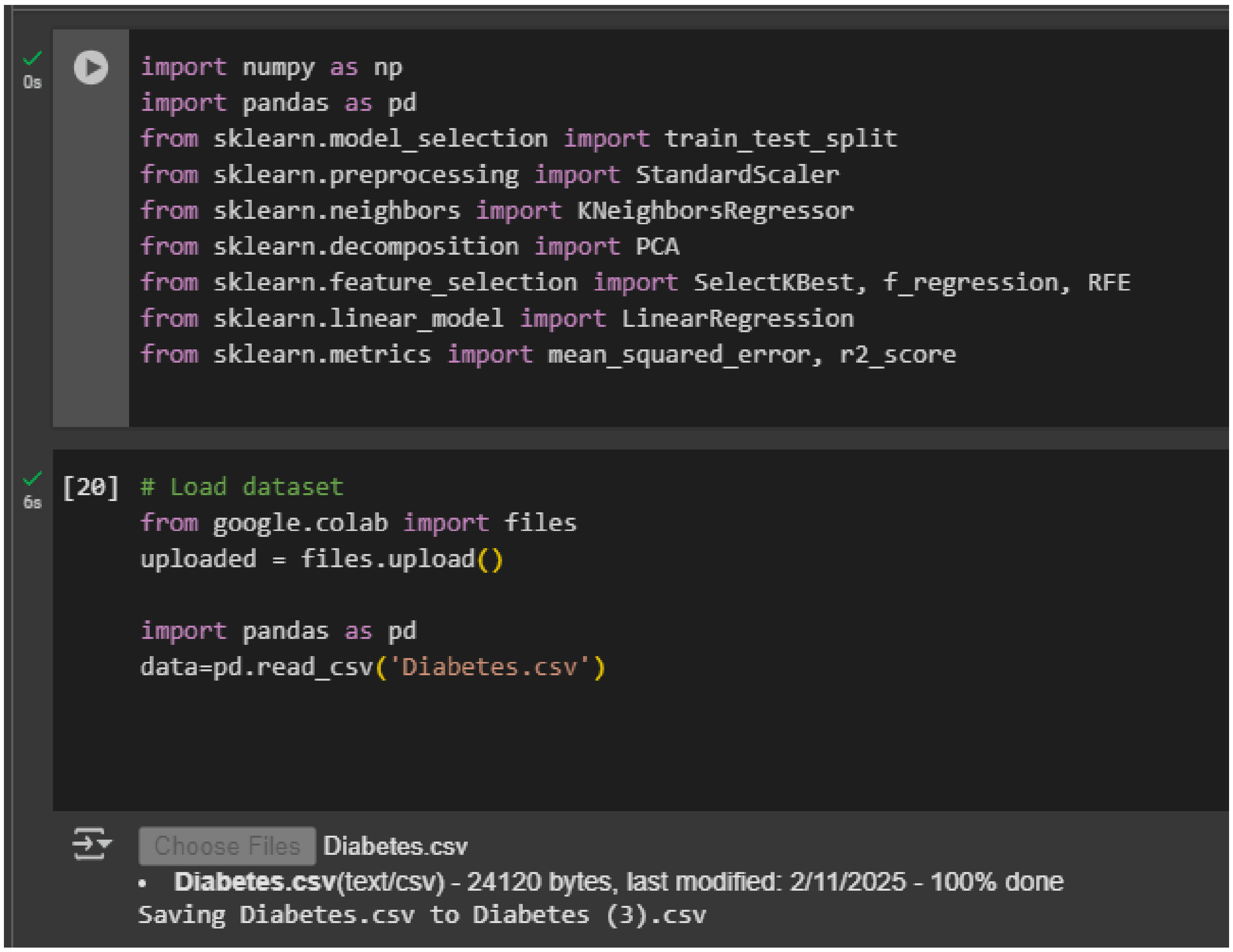
The goal of this lab is to implement a K-Nearest Neighbors (KNN) Regressor to predict Plasma Glucose Concentration in an Oral Glucose Tolerance Test using the Diabetes dataset. The performance of the KNN model is evaluated with different feature selection techniques and Principal Component Analysis (PCA).

The following steps are performed:

1. Data Preprocessing:
   - Splitting the dataset into training and testing sets.
   - Standardizing the features to improve model performance.
2. Feature Selection Techniques:
   - SelectKBest (Statistical Method using ANOVA F-score).
   - Recursive Feature Elimination (RFE) (Feature ranking using a Linear Regression model).
3. Dimensionality Reduction:
   - PCA (Principal Component Analysis) is used to transform the dataset into a reduced feature space.
4. Training KNN Models using selected features from the above techniques.

5. Performance Evaluation:

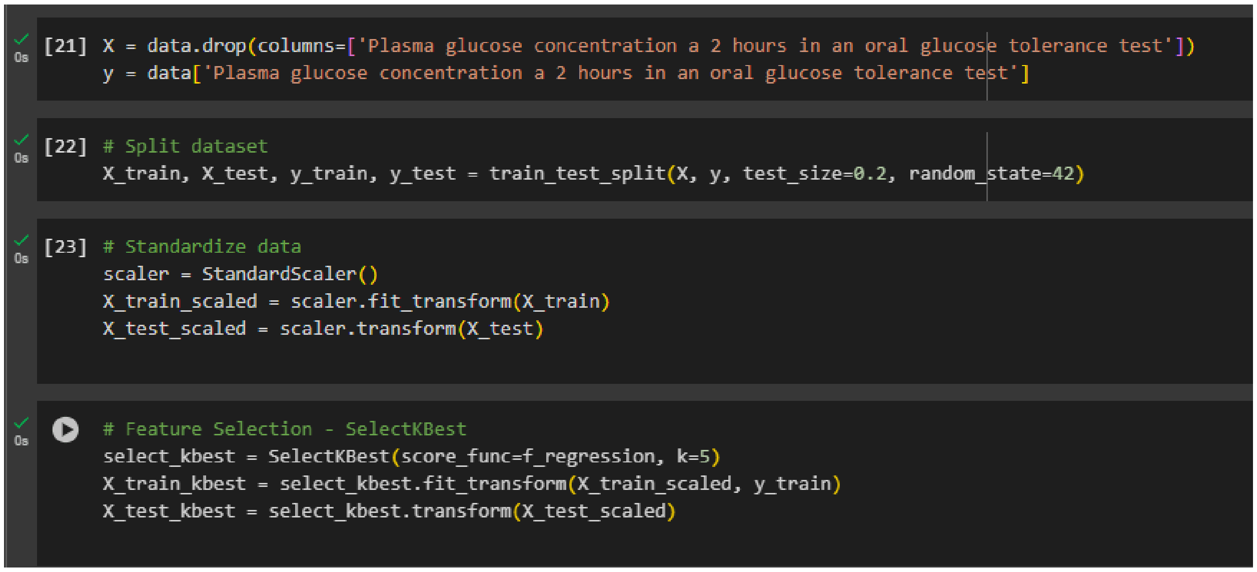
   - Comparing models using Root Mean Squared Error (RMSE) and R^2 Score to determine the best approach.



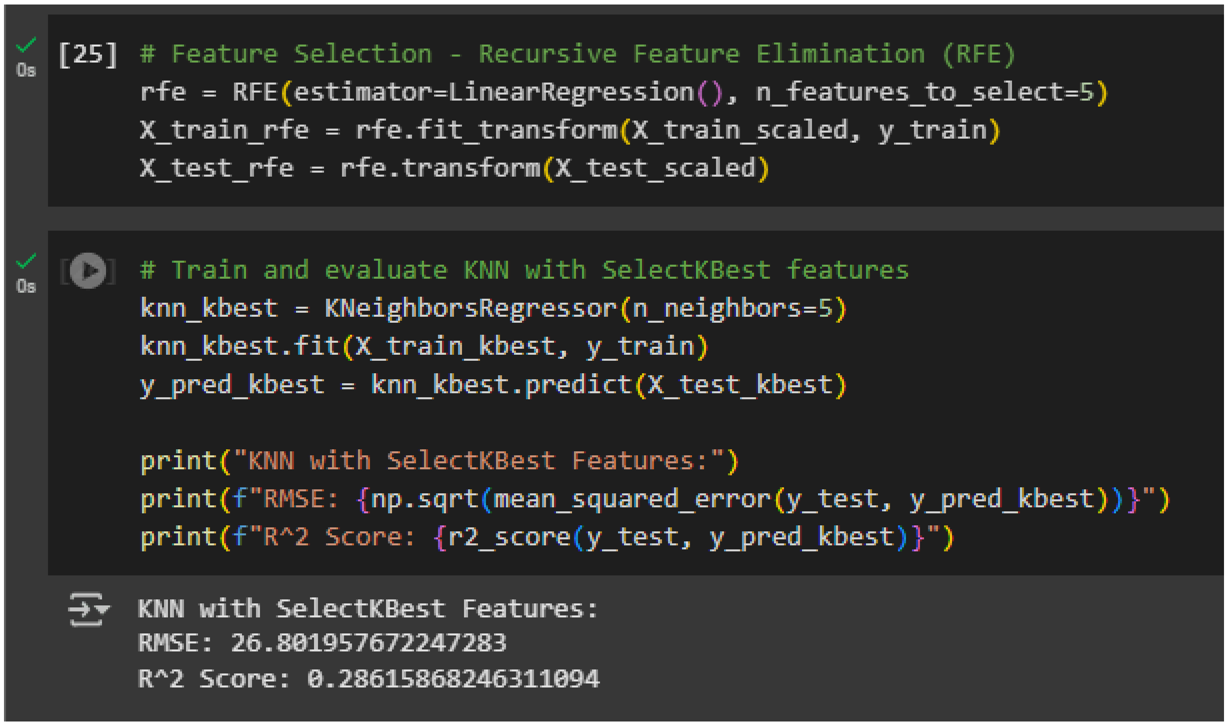
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsRegressor
from sklearn.decomposition import PCA
from sklearn.feature_selection import SelectKBest, f_regression, RFE
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

```
[20] # Load dataset
     from google.colab import files
     uploaded = files.upload()

     import pandas as pd
     data=pd.read_csv('Diabetes.csv')
```

```
Choose Files  Diabetes.csv
• Diabetes.csv(text/csv) - 24120 bytes, last modified: 2/11/2025 - 100% done
Saving Diabetes.csv to Diabetes (3).csv
```

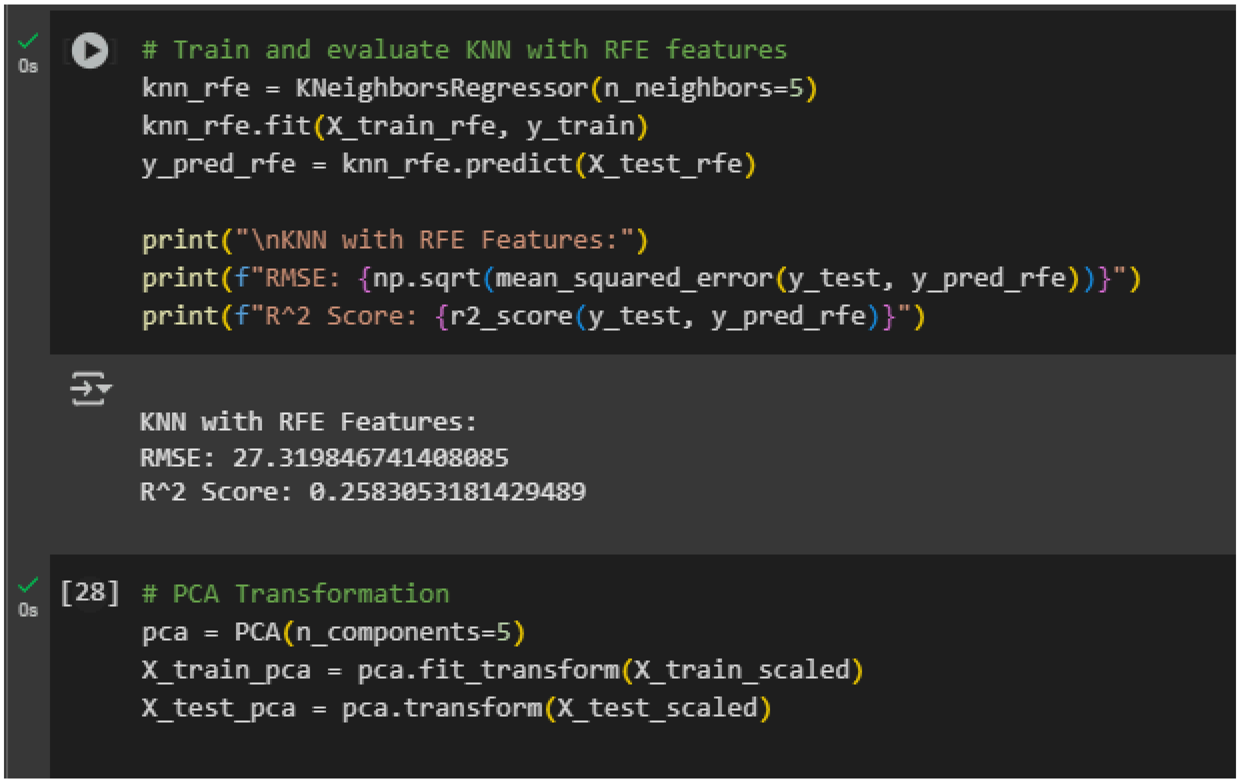


```
[21] X = data.drop(columns=['Plasma glucose concentration a 2 hours in an oral glucose tolerance test'])
     y = data['Plasma glucose concentration a 2 hours in an oral glucose tolerance test']

[22] # Split dataset
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

[23] # Standardize data
     scaler = StandardScaler()
     X_train_scaled = scaler.fit_transform(X_train)
     X_test_scaled = scaler.transform(X_test)

     # Feature Selection - SelectKBest
     select_kbest = SelectKBest(score_func=f_regression, k=5)
     X_train_kbest = select_kbest.fit_transform(X_train_scaled, y_train)
     X_test_kbest = select_kbest.transform(X_test_scaled)
```



```
[25] # Feature Selection - Recursive Feature Elimination (RFE)
     rfe = RFE(estimator=LinearRegression(), n_features_to_select=5)
     X_train_rfe = rfe.fit_transform(X_train_scaled, y_train)
     X_test_rfe = rfe.transform(X_test_scaled)

     # Train and evaluate KNN with SelectKBest features
     knn_kbest = KNeighborsRegressor(n_neighbors=5)
     knn_kbest.fit(X_train_kbest, y_train)
     y_pred_kbest = knn_kbest.predict(X_test_kbest)

     print("KNN with SelectKBest Features:")
     print(f"RMSE: {np.sqrt(mean_squared_error(y_test, y_pred_kbest))}")
     print(f"R^2 Score: {r2_score(y_test, y_pred_kbest)}")
```

```
KNN with SelectKBest Features:
RMSE: 26.801957672247283
R^2 Score: 0.28615868246311094
```

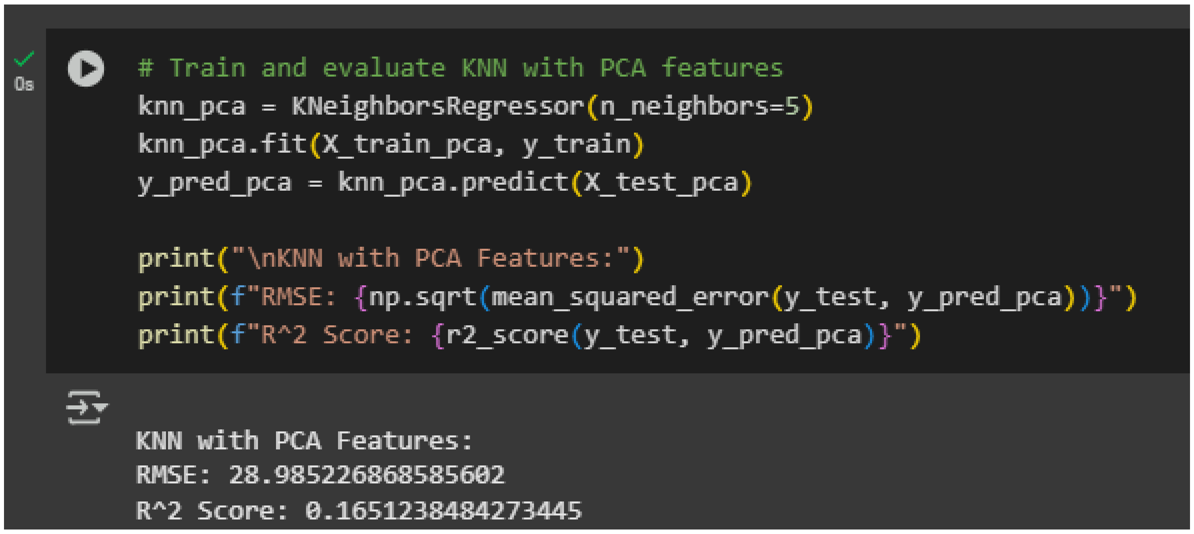


```
# Train and evaluate KNN with RFE features
knn_rfe = KNeighborsRegressor(n_neighbors=5)
knn_rfe.fit(X_train_rfe, y_train)
y_pred_rfe = knn_rfe.predict(X_test_rfe)

print("\nKNN with RFE Features:")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test, y_pred_rfe))}")
print(f"R^2 Score: {r2_score(y_test, y_pred_rfe)}")
```

```

KNN with RFE Features:
RMSE: 27.319846741408085
R^2 Score: 0.2583053181429489
```

[28]

```
# PCA Transformation
pca = PCA(n_components=5)
X_train_pca = pca.fit_transform(X_train_scaled)
X_test_pca = pca.transform(X_test_scaled)
```



```
# Train and evaluate KNN with PCA features
knn_pca = KNeighborsRegressor(n_neighbors=5)
knn_pca.fit(X_train_pca, y_train)
y_pred_pca = knn_pca.predict(X_test_pca)

print("\nKNN with PCA Features:")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test, y_pred_pca))}")
print(f"R^2 Score: {r2_score(y_test, y_pred_pca)}")
```

```

KNN with PCA Features:
RMSE: 28.985226868585602
R^2 Score: 0.1651238484273445
```

CONCLUSION:

In this lab, we implemented KNN Regression with different feature selection techniques and PCA to predict Plasma Glucose Concentration. The results highlight the importance of choosing the right features for better model performance. Depending on the dataset, feature selection methods like SelectKBest or RFE may improve accuracy, while PCA is useful when reducing dimensionality is a priority. From the analysis, we conclude that the best feature selection method is the one that achieves the lowest RMSE and highest R^2 Score, ensuring an optimal balance between complexity and predictive power.