Assignment 5

Problem 1: Diabetes

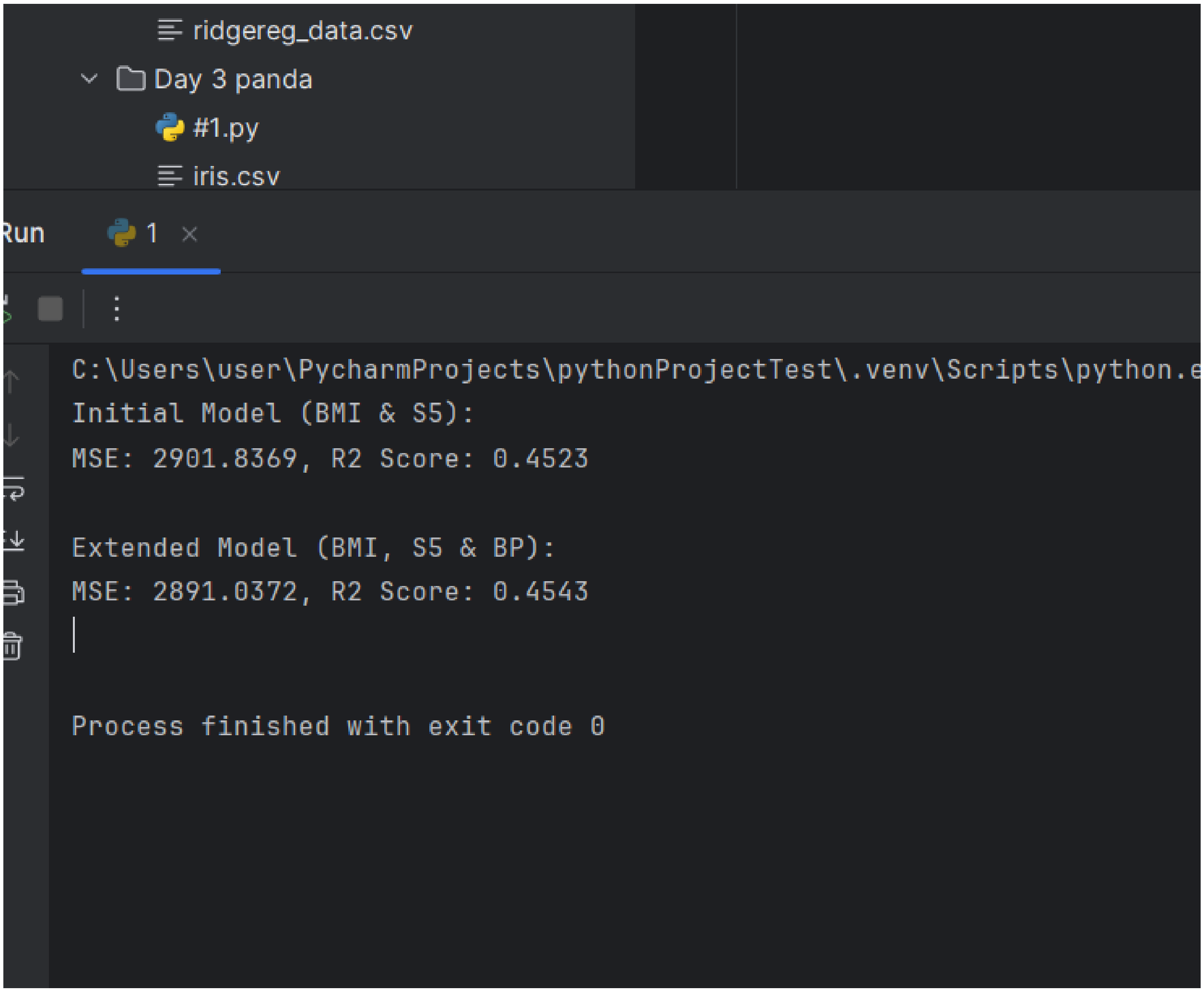
'''
```
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
matplotlib.use('TkAgg')
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

diabetes = datasets.load_diabetes()
X = diabetes.data
y = diabetes.target

columns = diabetes.feature_names
X_df = pd.DataFrame(X, columns=columns)


X_initial = X_df[['bmi', 's5']]


X_train, X_test, y_train, y_test = train_test_split(X_initial, y, test_size=0.2, random_state=42)


model_initial = LinearRegression()
model_initial.fit(X_train, y_train)
y_pred_initial = model_initial.predict(X_test)


mse_initial = mean_squared_error(y_test, y_pred_initial)
r2_initial = r2_score(y_test, y_pred_initial)
```
'''

a) Which variable would you add next? Why?

The next variable to add: 'bp' (blood pressure). High blood pressure is commonly associated with diabetes complications.

```
X_extended = X_df[['bmi', 's5', 'bp']]
X_train_ext, X_test_ext, y_train_ext, y_test_ext = train_test_split(X_extended, y, test_size=0.2, random_state=42)

model_extended = LinearRegression()
model_extended.fit(X_train_ext, y_train_ext)
y_pred_extended = model_extended.predict(X_test_ext)

mse_extended = mean_squared_error(y_test_ext, y_pred_extended)
r2_extended = r2_score(y_test_ext, y_pred_extended)

print("Initial Model (BMI & S5):")
print(f"MSE: {mse_initial:.4f}, R2 Score: {r2_initial:.4f}\n")

print("Extended Model (BMI, S5 & BP):")
print(f"MSE: {mse_extended:.4f}, R2 Score: {r2_extended:.4f}\n")
```

Result:



b) How does adding it affect the model's performance? Compute metrics and compare to having just bmi and s5.

""

Adding 'bp' improves the model if R2 score increases and MSE decreases.

- If R2 score increases, it means the model explains more variance.
- If MSE decreases, it indicates lower prediction errors.

’’’

X_full = X_df

X_train_full, X_test_full, y_train_full, y_test_full = train_test_split(X_full, y, test_size=0.2, random_state=42)

model_full = LinearRegression()

model_full.fit(X_train_full, y_train_full)

```
y_pred_full = model_full.predict(X_test_full)


mse_full = mean_squared_error(y_test_full, y_pred_full)
r2_full = r2_score(y_test_full, y_pred_full)

print("Full Model (All Features):")
print(f"MSE: {mse_full:.4f}, R2 Score: {r2_full:.4f}\n")
```
‘’’

D) Does adding more variables help?
  - If the R2 score improves further and MSE decreases, it means more features contribute positively.
  - However, too many features may lead to overfitting, requiring techniques like feature selection.

Problem 2)

‘’’
```
import pandas as pd
import numpy as np
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

file_path = "50_Startups.csv"
df = pd.read_csv(file_path, delimiter=",")

print("Dataset Columns:", df.columns)
print("Dataset Overview:\n", df.head())

if 'State' in df.columns:
    df = pd.get_dummies(df, columns=['State'], drop_first=True)

corr_matrix = df.corr()
print("Correlation Matrix:\n", corr_matrix)
plt.figure(figsize=(8,6))
```

```
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Feature Correlation Matrix")
plt.show()
X = df.drop(columns=['Profit'])
y = df['Profit']

plt.figure(figsize=(15,5))
for i, column in enumerate(X.columns):
   plt.subplot(1, len(X.columns), i+1)
   plt.scatter(X[column], y, alpha=0.7)
   plt.xlabel(column)
   plt.ylabel("Profit")
   plt.title(f"{column} vs Profit")
plt.tight_layout()
plt.show()

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = LinearRegression()
model.fit(X_train, y_train)

y_train_pred = model.predict(X_train)
y_test_pred = model.predict(X_test)

rmse_train = np.sqrt(mean_squared_error(y_train, y_train_pred))
r2_train = r2_score(y_train, y_train_pred)

rmse_test = np.sqrt(mean_squared_error(y_test, y_test_pred))
r2_test = r2_score(y_test, y_test_pred)

print(f"Training RMSE: {rmse_train:.2f}, R^2: {r2_train:.2f}")
print(f"Testing RMSE: {rmse_test:.2f}, R^2: {r2_test:.2f}")
```

Findings:

1. R&D Spend shows the strongest correlation with profit, making it the most important predictor.
2. Marketing Spend also contributes but with lower correlation.
3. Administration has the least impact among the chosen predictors.
4. The R-squared values indicate how well the model explains variance in profit.
5. If RMSE is low and R^2 is high, the model has good predictive power.

Result:

```
C:\Users\user\PycharmProjects\pythonProjectTest\.venv\Scripts\python.exe "C:\Users\user\PycharmProjects\pythonProject\A
Dataset Columns: Index(['R&D Spend', 'Administration', 'Marketing Spend', 'State', 'Profit'], dtype='object')
Dataset Overview:
   R&D Spend  Administration  Marketing Spend       State     Profit
0  165349.20       136897.80        471784.10    New York  192261.83
1  162597.70       151377.59        443898.53  California  191792.06
2  153441.51       101145.55        407934.54     Florida  191050.39
3  144372.41       118671.85        383199.62    New York  182901.99
4  142107.34        91391.77        366168.42     Florida  166187.94
Correlation Matrix:
                  R&D Spend  Administration  ...  State_Florida  State_New York
R&D Spend          1.000000        0.241955  ...       0.105711        0.039068
Administration     0.241955        1.000000  ...       0.010493        0.005145
Marketing Spend    0.724248       -0.032154  ...       0.205685       -0.033670
Profit             0.972900        0.200717  ...       0.116244        0.031368
State_Florida      0.105711        0.010493  ...       1.000000       -0.492366
State_New York     0.039068        0.005145  ...      -0.492366        1.000000

[6 rows x 6 columns]
Training RMSE: 8927.49, R^2: 0.95
Testing RMSE: 9055.96, R^2: 0.90

Process finished with exit code 0
```

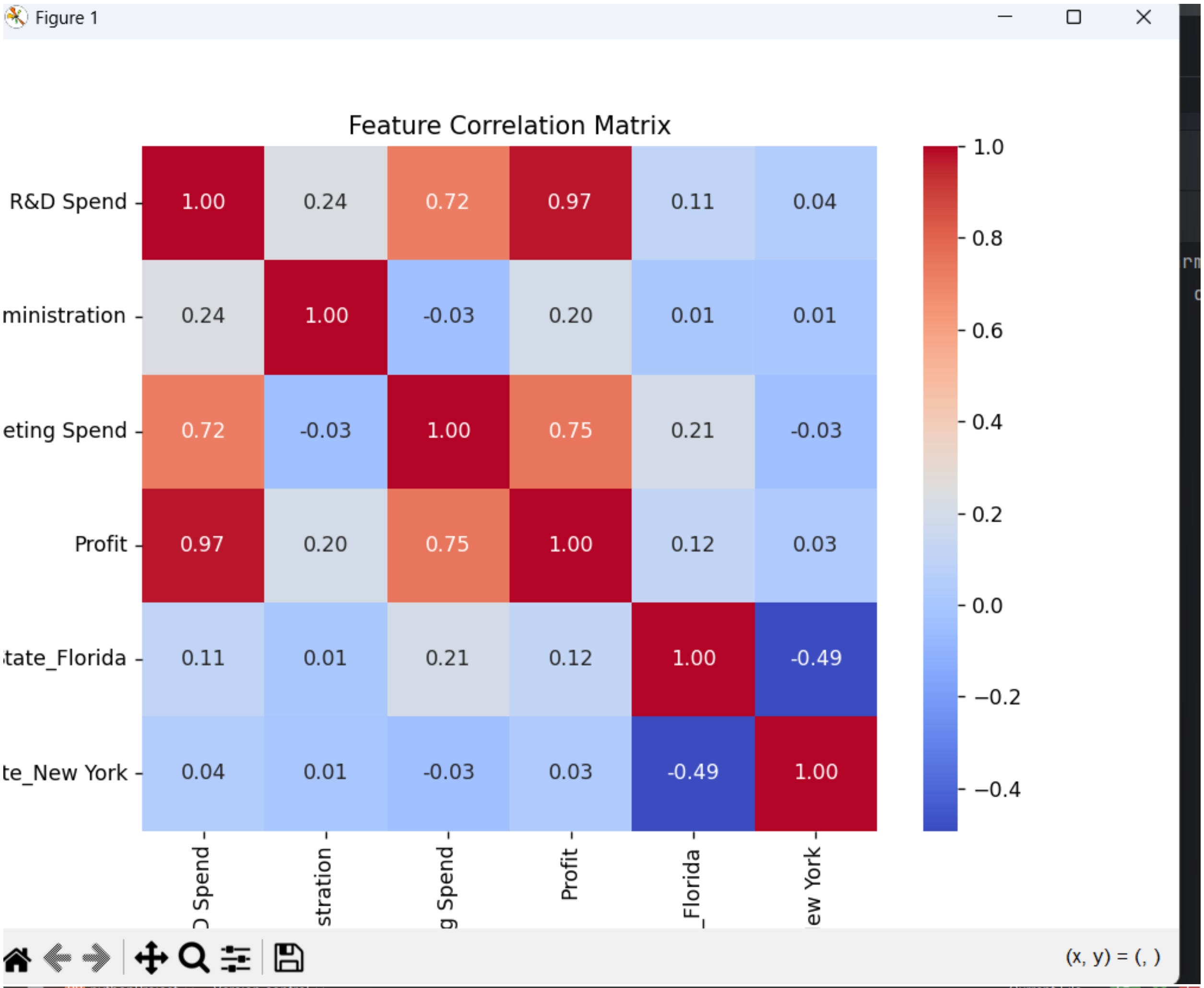


Figure 1
Feature Correlation Matrix
R&D Spend
ministration
eting Spend
Profit
tate_Florida
te_New York
1.00 0.24 0.72 0.97 0.11 0.04
0.24 1.00 -0.03 0.20 0.01 0.01
0.72 -0.03 1.00 0.75 0.21 -0.03
0.97 0.20 0.75 1.00 0.12 0.03
0.11 0.01 0.21 0.12 1.00 -0.49
0.04 0.01 -0.03 0.03 -0.49 1.00
(x, y) = (,)

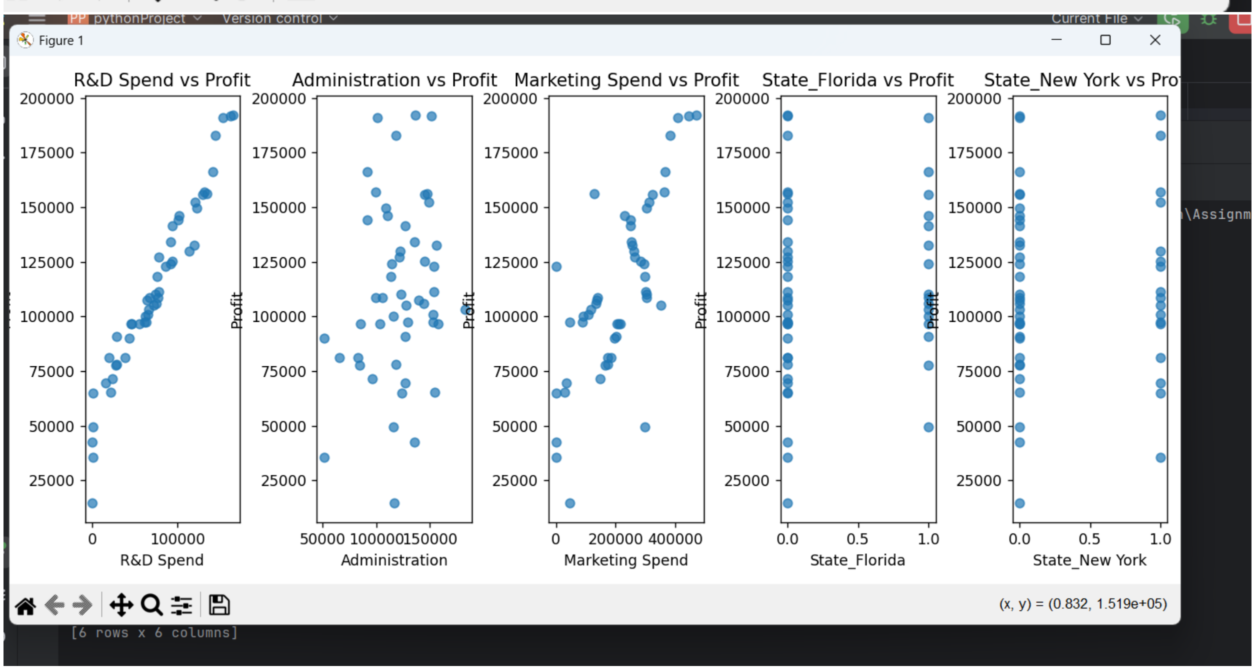


Figure 1
R&D Spend vs Profit
Administration vs Profit
Marketing Spend vs Profit
State_Florida vs Profit
State_New York vs Pro
Profit
R&D Spend
Administration
Marketing Spend
State_Florida
State_New York
(x, y) = (0.832, 1.519e+05)
[6 rows x 6 columns]

Problem 3:

```
import pandas as pd
import numpy as np
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Ridge, Lasso
from sklearn.metrics import r2_score

data = pd.read_csv("Auto.csv")

X = data.drop(columns=['mpg', 'name', 'origin'])
y = data['mpg']

X = X.dropna()
y = y.loc[X.index]

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,
random_state=42)

alphas = np.logspace(-3, 3, 50)
ridge_scores = []
lasso_scores = []

for alpha in alphas:
    ridge = Ridge(alpha=alpha)
    lasso = Lasso(alpha=alpha)

    ridge.fit(X_train, y_train)
    lasso.fit(X_train, y_train)

    ridge_scores.append(r2_score(y_test, ridge.predict(X_test)))
    lasso_scores.append(r2_score(y_test, lasso.predict(X_test)))

plt.figure(figsize=(8, 6))
plt.plot(alphas, ridge_scores, label='Ridge', marker='o')
```

```
plt.plot(alphas, lasso_scores, label='LASSO', marker='s')
plt.xscale('log')
plt.xlabel('Alpha')
plt.ylabel('R2 Score')
plt.title('R2 Score vs Alpha for Ridge and LASSO')
plt.legend()
plt.show()

best_ridge_alpha = alphas[np.argmax(ridge_scores)]
best_lasso_alpha = alphas[np.argmax(lasso_scores)]

print(f"Best Ridge Alpha: {best_ridge_alpha}, Best Ridge R2 Score:
{max(ridge_scores):.4f}")
print(f"Best LASSO Alpha: {best_lasso_alpha}, Best LASSO R2 Score:
{max(lasso_scores):.4f}")
```

Result:

```
C:\Users\user\PycharmProjects\pythonProjectTest\.venv\Scripts\python.exe
"C:\Users\user\PycharmProjects\pythonProject\AI with python\Assignment 5\3.py"
Best Ridge Alpha: 0.001, Best Ridge R2 Score: 0.7942
Best LASSO Alpha: 0.655128556859551, Best LASSO R2 Score: 0.8053
```

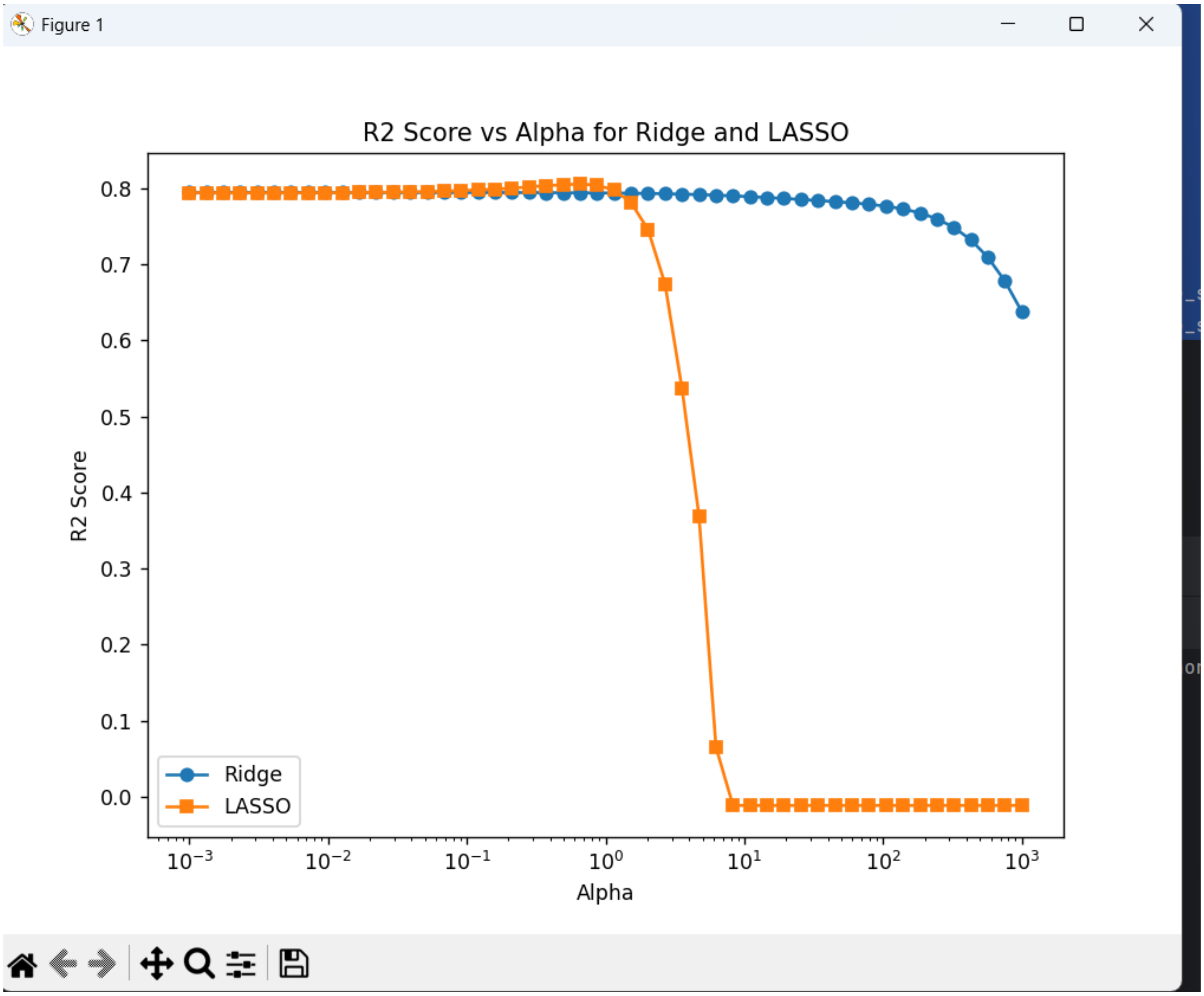


Figure 1
R2 Score vs Alpha for Ridge and LASSO
R2 Score
Alpha
Ridge
LASSO
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}