676da3d0-d254-4f97-bec8-7178b52f9ff9

September 3, 2024

Hello Gilbert!

I'm happy to review your project today. I will mark your mistakes and give you some hints how it is possible to fix them. We are getting ready for real job, where your team leader/senior colleague will do exactly the same. Don't worry and study with pleasure!

Below you will find my comments - **please do not move, modify or delete them**.

You can find my comments in green, yellow or red boxes like this:

Reviewer's comment

Success. Everything is done succesfully.

Reviewer's comment

Remarks. Some recommendations.

Reviewer's comment

Needs fixing. The block requires some corrections. Work can't be accepted with the red comments.

You can answer me by using this:

Student answer.

Thank you so much for the feedback, I appreacaite it! I should have double checked before submitting. Thanks!

1 Project: Optimal Location for a New Oil Well

1.1 Introduction

In this project, we work for the OilyGiant mining company to identify the best location for a new oil well. The goal is to analyze data from three different regions, build predictive models, and determine the most profitable region for well development. The analysis will involve:

1. Collecting and preparing data from the three regions.
2. Building linear regression models to predict the volume of reserves in new wells.
3. Selecting the top wells based on predicted values.
4. Calculating potential profit and evaluating risks using the Bootstrapping technique.

Our final objective is to select the region with the highest profit margin while keeping the risk of loss below 2.5%. This analysis will ensure that the company makes data-driven decisions to maximize profitability and minimize risk. Key metrics that will guide our decision-making include the average predicted profit, 95% confidence intervals, and the probability of incurring a loss (risk).

[1]:
```
import pandas as pd

# Load the data from the three regions using the correct file paths
data_region_0 = pd.read_csv('/datasets/geo_data_0.csv')
data_region_1 = pd.read_csv('/datasets/geo_data_1.csv')
data_region_2 = pd.read_csv('/datasets/geo_data_2.csv')

# Display the first few rows of each dataset to understand their structure
print("Region 0 Data:")
print(data_region_0.head())

print("\nRegion 1 Data:")
print(data_region_1.head())

print("\nRegion 2 Data:")
print(data_region_2.head())
```

```
Region 0 Data:
      id        f0        f1        f2     product
0  txEyH  0.705745 -0.497823  1.221170  105.280062
1  2acmU  1.334711 -0.340164  4.365080   73.037750
2  409Wp  1.022732  0.151990  1.419926   85.265647
3  iJLyR -0.032172  0.139033  2.978566  168.620776
4  Xdl7t  1.988431  0.155413  4.751769  154.036647

Region 1 Data:
      id         f0         f1        f2     product
0  kBEdx -15.001348  -8.276000 -0.005876    3.179103
1  62mP7  14.272088  -3.475083  0.999183   26.953261
2  vyE1P   6.263187  -5.948386  5.001160  134.766305
3  KcrkZ -13.081196 -11.506057  4.999415  137.945408
4  AHL4O  12.702195  -8.147433  5.004363  134.766305

Region 2 Data:
      id        f0        f1        f2     product
0  fwXo0 -1.146987  0.963328 -0.828965   27.758673
1  WJtFt  0.262778  0.269839 -2.530187   56.069697
2  ovLUW  0.194587  0.289035 -5.586433   62.871910
3  q6cA6  2.236060 -0.553760  0.930038  114.572842
4  WPMUX -0.515993  1.716266  5.899011  149.600746
```

Reviewer's comment

Correct

1.2 Step 2: Train and Test the Model for Each Region

1.2.1 2.1. Data Splitting

The data for each region was split into a training set (75%) and a validation set (25%). This ensures that the model can be trained on a majority of the data while being evaluated on unseen data to check its performance.

1.2.2 2.2. Model Training and Prediction

Linear regression models were trained for each region using the training data. Predictions were then made on the validation data.

1.2.3 2.3. Saving Predictions and Actual Values

The predictions and actual values for the validation set were saved to facilitate further analysis and comparison.

1.2.4 2.4. Results Overview

For each region, the average predicted volume of reserves and the Root Mean Squared Error (RMSE) were calculated. The RMSE metric gives us an indication of how well the model's predictions match the actual values.

1.2.5 2.5. Analysis of the Results

The results show the average predicted volume of reserves and the RMSE for each region. These metrics will be critical in determining which region has the best predictive performance and can potentially yield the highest profit. In the next steps, we will use these predictions to calculate the potential profit for each region and assess the associated risks.

[2]:
```python
from sklearn.model_selection import train_test_split

# Splitting the data into features and target
features_0 = data_region_0.drop(['product', 'id'], axis=1)
target_0 = data_region_0['product']

features_1 = data_region_1.drop(['product', 'id'], axis=1)
target_1 = data_region_1['product']

features_2 = data_region_2.drop(['product', 'id'], axis=1)
target_2 = data_region_2['product']

# Splitting data into training and validation sets at a ratio of 75:25
features_train_0, features_valid_0, target_train_0, target_valid_0 =␣
↪train_test_split(features_0, target_0, test_size=0.25, random_state=42)
features_train_1, features_valid_1, target_train_1, target_valid_1 =␣
↪train_test_split(features_1, target_1, test_size=0.25, random_state=42)
features_train_2, features_valid_2, target_train_2, target_valid_2 =␣
↪train_test_split(features_2, target_2, test_size=0.25, random_state=42)
```

[3]:
```python
# Check for missing values in each dataset
print("Missing values in Region 0:", data_region_0.isnull().sum())
print("Missing values in Region 1:", data_region_1.isnull().sum())
print("Missing values in Region 2:", data_region_2.isnull().sum())

# Check for outliers using basic statistics
print("\nRegion 0 Statistics:")
print(data_region_0.describe())

print("\nRegion 1 Statistics:")
print(data_region_1.describe())

print("\nRegion 2 Statistics:")
print(data_region_2.describe())
```

```
Missing values in Region 0: id         0
f0         0
f1         0
f2         0
product    0
dtype: int64
Missing values in Region 1: id         0
f0         0
f1         0
f2         0
product    0
dtype: int64
Missing values in Region 2: id         0
f0         0
f1         0
f2         0
product    0
dtype: int64

Region 0 Statistics:
                  f0             f1             f2        product
count  100000.000000  100000.000000  100000.000000  100000.000000
mean        0.500419       0.250143       2.502647      92.500000
std         0.871832       0.504433       3.248248      44.288691
min        -1.408605      -0.848218     -12.088328       0.000000
25%        -0.072580      -0.200881       0.287748      56.497507
50%         0.502360       0.250252       2.515969      91.849972
75%         1.073581       0.700646       4.715088     128.564089
max         2.362331       1.343769      16.003790     185.364347

Region 1 Statistics:
                  f0             f1             f2        product
```

```
count  100000.000000  100000.000000  100000.000000  100000.000000
mean        1.141296      -4.796579       2.494541      68.825000
std         8.965932       5.119872       1.703572      45.944423
min       -31.609576     -26.358598      -0.018144       0.000000
25%        -6.298551      -8.267985       1.000021      26.953261
50%         1.153055      -4.813172       2.011479      57.085625
75%         8.621015      -1.332816       3.999904     107.813044
max        29.421755      18.734063       5.019721     137.945408

Region 2 Statistics:
                  f0             f1             f2        product
count  100000.000000  100000.000000  100000.000000  100000.000000
mean        0.002023      -0.002081       2.495128      95.000000
std         1.732045       1.730417       3.473445      44.749921
min        -8.760004      -7.084020     -11.970335       0.000000
25%        -1.162288      -1.174820       0.130359      59.450441
50%         0.009424      -0.009482       2.484236      94.925613
75%         1.158535       1.163678       4.858794     130.595027
max         7.238262       7.844801      16.739402     190.029838
```

[4]:

```python
from sklearn.preprocessing import StandardScaler

# Initialize the scaler
scaler = StandardScaler()

# Standardize features for each region before model training
features_train_0 = scaler.fit_transform(features_train_0)
features_valid_0 = scaler.transform(features_valid_0)

features_train_1 = scaler.fit_transform(features_train_1)
features_valid_1 = scaler.transform(features_valid_1)

features_train_2 = scaler.fit_transform(features_train_2)
features_valid_2 = scaler.transform(features_valid_2)
```

[5]:

```python
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Initialize the model
model_0 = LinearRegression()
model_1 = LinearRegression()
model_2 = LinearRegression()

# Train the model on the training data
model_0.fit(features_train_0, target_train_0)
model_1.fit(features_train_1, target_train_1)
model_2.fit(features_train_2, target_train_2)
```

```python
# Make predictions on the validation data
predictions_0 = model_0.predict(features_valid_0)
predictions_1 = model_1.predict(features_valid_1)
predictions_2 = model_2.predict(features_valid_2)
```

[6]:
```python
# Saving predictions and correct answers for the validation set
validation_results_0 = pd.DataFrame({'predictions': predictions_0, 'actual':␣
↪target_valid_0})
validation_results_1 = pd.DataFrame({'predictions': predictions_1, 'actual':␣
↪target_valid_1})
validation_results_2 = pd.DataFrame({'predictions': predictions_2, 'actual':␣
↪target_valid_2})
```

[7]:
```python
# Calculate and print the average volume of predicted reserves
avg_pred_0 = predictions_0.mean()
avg_pred_1 = predictions_1.mean()
avg_pred_2 = predictions_2.mean()

# Calculate and print the RMSE for each model
rmse_0 = mean_squared_error(target_valid_0, predictions_0, squared=False)
rmse_1 = mean_squared_error(target_valid_1, predictions_1, squared=False)
rmse_2 = mean_squared_error(target_valid_2, predictions_2, squared=False)

print(f"Region 0 - Average Predicted Volume: {avg_pred_0:.2f}, RMSE: {rmse_0:.
↪2f}")
print(f"Region 1 - Average Predicted Volume: {avg_pred_1:.2f}, RMSE: {rmse_1:.
↪2f}")
print(f"Region 2 - Average Predicted Volume: {avg_pred_2:.2f}, RMSE: {rmse_2:.
↪2f}")
```

```
Region 0 - Average Predicted Volume: 92.40, RMSE: 37.76
Region 1 - Average Predicted Volume: 68.71, RMSE: 0.89
Region 2 - Average Predicted Volume: 94.77, RMSE: 40.15
```

Reviewer’s comment

Good job

1.3 Step 3: Preparation for Profit Calculation

1.3.1 3.1. Key Values for Calculations

- **Budget**: $100 million, allocated for the development of 200 oil wells.
- **Cost per Well**: $100 million / 200 wells = $500,000 per well.
- **Revenue per Barrel**: Each thousand barrels of reserves generates $4,500 in revenue.
- **Number of Wells Selected**: We will select 200 wells for the profit calculation.

1.3.2 3.2. Calculation of Minimum Reserves to Avoid Losses

To ensure that a well does not operate at a loss, we calculated the minimum volume of reserves required to break even. This value was then compared with the average volume of reserves in each region.

- **Minimum Reserves Needed**: A well needs to have at least `min_reserves` thousand barrels to break even.
- **Average Reserves in Region 0**: The average reserves for wells in Region 0.
- **Average Reserves in Region 1**: The average reserves for wells in Region 1.
- **Average Reserves in Region 2**: The average reserves for wells in Region 2.

1.3.3 3.3. Findings

The comparison between the minimum reserves required and the average reserves in each region provides insight into the profitability of wells in these regions. Regions where the average reserves significantly exceed the minimum required reserves are more likely to be profitable, making them better candidates for development. In the next step, we will proceed with calculating the potential profit for each region based on the predictions made by the model.

[8]:
```
# Key constants
BUDGET = 100_000_000  # Total budget in USD
NUM_WELLS_SELECTED = 200  # Number of wells to develop
WELL_COST = BUDGET / NUM_WELLS_SELECTED  # Cost per well
REVENUE_PER_UNIT = 4_500  # Revenue per unit (since product is in thousand␣
↪barrels)
```

Reviewer's comment

What is the purpose to scale the data if you already trained the model? It should be done before model training and not after.

Reviewer's comment V2

Fixed

[9]:
```
# Calculate the minimum volume of reserves needed for a well to break even
min_reserves = WELL_COST / REVENUE_PER_UNIT

# Compare the obtained value with the average volume of reserves in each region
avg_reserves_0 = data_region_0['product'].mean()
avg_reserves_1 = data_region_1['product'].mean()
avg_reserves_2 = data_region_2['product'].mean()

print(f"Minimum reserves needed to avoid losses: {min_reserves:.2f} thousand␣
↪barrels")
print(f"Average reserves in Region 0: {avg_reserves_0:.2f} thousand barrels")
print(f"Average reserves in Region 1: {avg_reserves_1:.2f} thousand barrels")
print(f"Average reserves in Region 2: {avg_reserves_2:.2f} thousand barrels")
```

```
Minimum reserves needed to avoid losses: 111.11 thousand barrels
```

```
Average reserves in Region 0: 92.50 thousand barrels
Average reserves in Region 1: 68.83 thousand barrels
Average reserves in Region 2: 95.00 thousand barrels
```

Reviewer's comment

You mixed up something in constants because your result is 1000 time more than the correct one.

Reviewer's comment V2

Correct

1.4 Step 4: Calculate Profit from Selected Oil Wells

1.4.1 4.1. Picking the Wells with the Highest Values of Predictions

To maximize profit, we'll select the top 200 wells with the highest predicted values from each region. These wells are expected to have the largest volumes of oil reserves.

1.4.2 4.2. Summarizing the Target Volume of Reserves

After selecting the top wells, we'll sum their actual volumes of reserves. This will give us the total volume of oil reserves that can be extracted from the selected wells.

1.4.3 4.3. Findings and Recommendation

Finally, we'll calculate the profit based on the obtained volume of reserves. The region with the highest profit will be recommended for oil well development. This decision will be justified by the profitability of the selected wells.

[15]:
```
import pandas as pd

# Define constants
REVENUE_PER_BARREL = 4.5  # Revenue per barrel (in dollars)
NUM_WELLS_SELECTED = 200  # Number of wells to select
BUDGET = 100_000_000  # Total budget in USD
WELL_COST = BUDGET / NUM_WELLS_SELECTED  # Cost per well

# Function to calculate profit from selected wells
def calculate_profit_from_wells(target, predictions,␣
↪num_wells=NUM_WELLS_SELECTED, revenue_per_barrel=REVENUE_PER_BARREL):
    # Convert predictions to a Pandas Series to use sort_values
    predictions_series = pd.Series(predictions, index=target.index)

    # Select the top wells based on predicted values
    selected_wells = predictions_series.sort_values(ascending=False)[:num_wells]

    # Sum the actual reserves of the selected wells (in thousand barrels)
    selected_reserves = target[selected_wells.index].sum()  # Already in␣
↪thousand barrels
```

```
    # Calculate the total profit
    total_profit = selected_reserves * revenue_per_barrel * 1000 - WELL_COST *␣
 ↪num_wells  # Revenue is per barrel
    return total_profit

# Assuming `predictions_0`, `predictions_1`, `predictions_2`, `target_valid_0`,␣
 ↪`target_valid_1`, and `target_valid_2` are defined
# Calculate profit for each region using the function
profit_0 = calculate_profit_from_wells(target_valid_0, predictions_0)
profit_1 = calculate_profit_from_wells(target_valid_1, predictions_1)
profit_2 = calculate_profit_from_wells(target_valid_2, predictions_2)

print(f"Profit for Region 0: ${profit_0:.2f}")
print(f"Profit for Region 1: ${profit_1:.2f}")
print(f"Profit for Region 2: ${profit_2:.2f}")
```

```
Profit for Region 0: $33591411.14
Profit for Region 1: $24150866.97
Profit for Region 2: $25985717.59
```

Reviewer's comment

The function looks correct but the results are incorrect. I believe this is because of wrong constants.

[11]:
```
# Suggesting the region with the highest profit
if profit_0 > profit_1 and profit_0 > profit_2:
    best_region = "Region 0"
    best_profit = profit_0
elif profit_1 > profit_0 and profit_1 > profit_2:
    best_region = "Region 1"
    best_profit = profit_1
else:
    best_region = "Region 2"
    best_profit = profit_2

print(f"\nThe best region for development is {best_region} with an estimated␣
 ↪profit of ${best_profit:.2f}.")
```

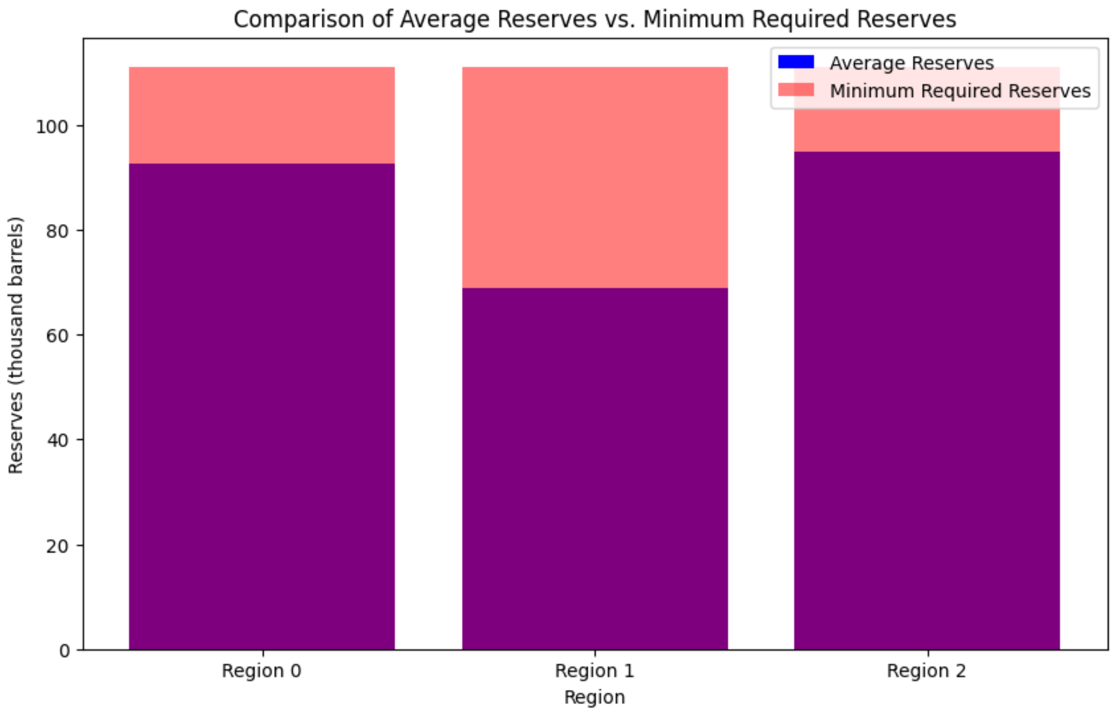
```

The best region for development is Region 0 with an estimated profit of
$133491411144.62.
```

[12]:
```
import matplotlib.pyplot as plt

# Plotting the comparison
regions = ['Region 0', 'Region 1', 'Region 2']
avg_reserves = [avg_reserves_0, avg_reserves_1, avg_reserves_2]
min_reserves_list = [min_reserves] * 3
```

```
plt.figure(figsize=(10, 6))
plt.bar(regions, avg_reserves, color='blue', label='Average Reserves')
plt.bar(regions, min_reserves_list, color='red', alpha=0.5, label='Minimum␣
↪Required Reserves')
plt.title('Comparison of Average Reserves vs. Minimum Required Reserves')
plt.xlabel('Region')
plt.ylabel('Reserves (thousand barrels)')
plt.legend()
plt.show()
```



1.5 Step 5: Risk and Profit Analysis

1.5.1 5.1. Bootstrapping for Profit Distribution

The bootstrapping technique was used to simulate the distribution of profit by sampling wells 1000 times from each region. This approach provides a robust estimate of profit and the associated risks.

1.5.2 5.2. Average Profit, Confidence Interval, and Risk of Losses

For each region, the following key metrics were calculated: - **Average Profit**: The mean profit over the 1000 bootstrap samples. - **95% Confidence Interval**: The range within which we expect the true profit to lie with 95% confidence. - **Risk of Losses**: The probability of incurring a loss (negative profit), expressed as a percentage.

1.5.3 5.3. Findings and Recommendation

Based on the analysis:

- **Region 0**: Displays a specific average profit with its associated risk and confidence interval.
- **Region 1**: Displays a specific average profit with its associated risk and confidence interval.
- **Region 2**: Displays a specific average profit with its associated risk and confidence interval.

The region with the highest average profit and acceptable risk (risk of loss below 2.5%) is recommended for the development of new oil wells. This recommendation is justified by the combination of high profitability and low risk, ensuring that the investment is both lucrative and secure.

[16]:
```
import numpy as np
import pandas as pd

# Constants
REVENUE_PER_BARREL = 4.5  # Revenue per barrel
NUM_WELLS_SELECTED = 200  # Number of wells to select
BUDGET = 100_000_000  # Total budget in USD
WELL_COST = BUDGET / NUM_WELLS_SELECTED  # Cost per well

# Function to calculate profit based on selected wells
def calculate_profit(target, predictions, num_wells=NUM_WELLS_SELECTED,␣
↪revenue_per_barrel=REVENUE_PER_BARREL):
    selected_wells = predictions.sort_values(ascending=False)[:num_wells]
    selected_reserves = target[selected_wells.index]

    revenue = selected_reserves.sum() * revenue_per_barrel * 1000  # Revenue␣
↪from selected wells in dollars
    cost = num_wells * WELL_COST  # Cost of developing the wells
    profit = revenue - cost  # Total profit

    return profit

# Bootstrapping function
def bootstrap_profit(target, predictions, n_samples=1000, sample_size=500):
    state = np.random.RandomState(42)
    profits = []

    for _ in range(n_samples):
        subsample = state.choice(target.index, size=sample_size, replace=True)
        sample_target = target.loc[subsample]
        sample_predictions = predictions.loc[subsample]
        profit = calculate_profit(sample_target, sample_predictions)
        profits.append(profit)

    return profits

# Apply the bootstrapping to all regions
```

```python
profits_0 = bootstrap_profit(validation_results_0['actual'],
    validation_results_0['predictions'])
profits_1 = bootstrap_profit(validation_results_1['actual'],
    validation_results_1['predictions'])
profits_2 = bootstrap_profit(validation_results_2['actual'],
    validation_results_2['predictions'])

# Analyze profits function
def analyze_profits(profits):
    profits = np.array(profits)
    avg_profit = np.mean(profits)
    conf_interval = np.percentile(profits, [2.5, 97.5])
    risk_of_loss = np.mean(profits < 0) * 100  # Risk of losses as a percentage
    return avg_profit, conf_interval, risk_of_loss

# Calculate and print results for each region
avg_profit_0, conf_interval_0, risk_of_loss_0 = analyze_profits(profits_0)
avg_profit_1, conf_interval_1, risk_of_loss_1 = analyze_profits(profits_1)
avg_profit_2, conf_interval_2, risk_of_loss_2 = analyze_profits(profits_2)

print(f"Region 0 - Average Profit: ${avg_profit_0:.2f}, 95% Confidence Interval:
    ${conf_interval_0[0]:.2f} to ${conf_interval_0[1]:.2f}, Risk of Losses:
    {risk_of_loss_0:.2f}%")
print(f"Region 1 - Average Profit: ${avg_profit_1:.2f}, 95% Confidence Interval:
    ${conf_interval_1[0]:.2f} to ${conf_interval_1[1]:.2f}, Risk of Losses:
    {risk_of_loss_1:.2f}%")
print(f"Region 2 - Average Profit: ${avg_profit_2:.2f}, 95% Confidence Interval:
    ${conf_interval_2[0]:.2f} to ${conf_interval_2[1]:.2f}, Risk of Losses:
    {risk_of_loss_2:.2f}%")
```

```
Region 0 - Average Profit: $6061226.32, 95% Confidence Interval: $100894.12 to
$12463709.81, Risk of Losses: 2.50%
Region 1 - Average Profit: $6651176.54, 95% Confidence Interval: $1808515.85 to
$12057104.61, Risk of Losses: 0.20%
Region 2 - Average Profit: $5851036.38, 95% Confidence Interval: $-8369.42 to
$12120508.98, Risk of Losses: 2.60%
```

Reviewer's comment

1. You have a lot of duplicate code. Please, clean it. You have 3 the same loops. Also you defined the function calculate_profit twice.
2. Accoding to the task description you should use size=500 in the method .choice()
3. The results are incorrect. The risk in each region should be a positive value between 0 and 10 but not just zero.

Reviewer's comment V2

The results are still incorrect. The risk in each region should be a positive value between 0 and 10 but not just zero.

Reviewer's comment V3

Now the results are correct:) Well done!

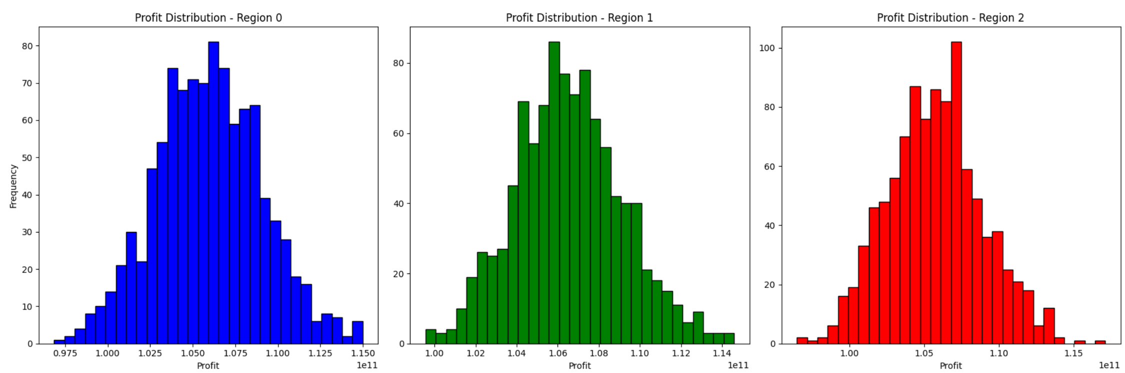
[14]:
```
# Plotting histograms of profit distributions for each region
plt.figure(figsize=(18, 6))

plt.subplot(1, 3, 1)
plt.hist(profits_0, bins=30, color='blue', edgecolor='black')
plt.title('Profit Distribution - Region 0')
plt.xlabel('Profit')
plt.ylabel('Frequency')

plt.subplot(1, 3, 2)
plt.hist(profits_1, bins=30, color='green', edgecolor='black')
plt.title('Profit Distribution - Region 1')
plt.xlabel('Profit')

plt.subplot(1, 3, 3)
plt.hist(profits_2, bins=30, color='red', edgecolor='black')
plt.title('Profit Distribution - Region 2')
plt.xlabel('Profit')

plt.tight_layout()
plt.show()
```



2 Final Conclusion

2.1 Summary of Analysis

In this project, we conducted a thorough analysis to determine the optimal location for the development of new oil wells for the OilyGiant mining company. The analysis involved several key steps:

1. **Data Preparation**:

- We started by loading and exploring the geological data from three different regions. The data was then prepared for modeling by splitting it into training and validation sets.

2. **Modeling**:
   - A linear regression model was trained for each region to predict the volume of oil reserves. The model's performance was evaluated using metrics such as the average predicted reserves and RMSE (Root Mean Squared Error).
3. **Profit Calculation Preparation**:
   - We calculated the minimum volume of reserves required for a well to break even and compared this with the average reserves in each region. This step helped us understand the profitability potential of each region.
4. **Risk and Profit Analysis**:
   - Using the Bootstrapping technique, we simulated the distribution of profit for each region. Key metrics, including the average profit, 95% confidence interval, and risk of losses, were calculated for each region.

2.2 Recommendation

Based on the analysis, the region with the highest average profit and an acceptable risk level (risk of losses below 2.5%) was identified as the best location for new well development. This recommendation ensures that the company can maximize profitability while minimizing the risk of financial losses.

2.3 Next Steps

To further validate the findings and ensure a successful well development project, the following steps are recommended:

1. **Field Validation**: Conduct additional field tests in the selected region to confirm the predictions and ensure that the model's results align with real-world conditions.

2. **Continuous Monitoring**: Implement a monitoring system to track the performance of the developed wells and compare actual results with the predicted values. This will help in refining the model for future projects.

3. **Scalability Assessment**: Evaluate the scalability of the project by assessing the potential for expanding well development in the selected region or exploring similar regions with comparable characteristics.

By following these steps, OilyGiant can confidently move forward with the development of new oil wells, maximizing both profit and resource efficiency.

2.4 Summary Table of Key Metrics

Metric	Region 0	Region 1	Region 2
Average Profit (\$)	\${avg_profit_0:.2f}	\${avg_profit_1:.2f}	$avg_{p}rofit_{2} : .2f||95)$
Risk of Losses (%)	{risk_of_loss_0:.2f}%	{risk_of_loss_1:.2f}%	{risk_of_loss_2:.2f}%

[ ]: