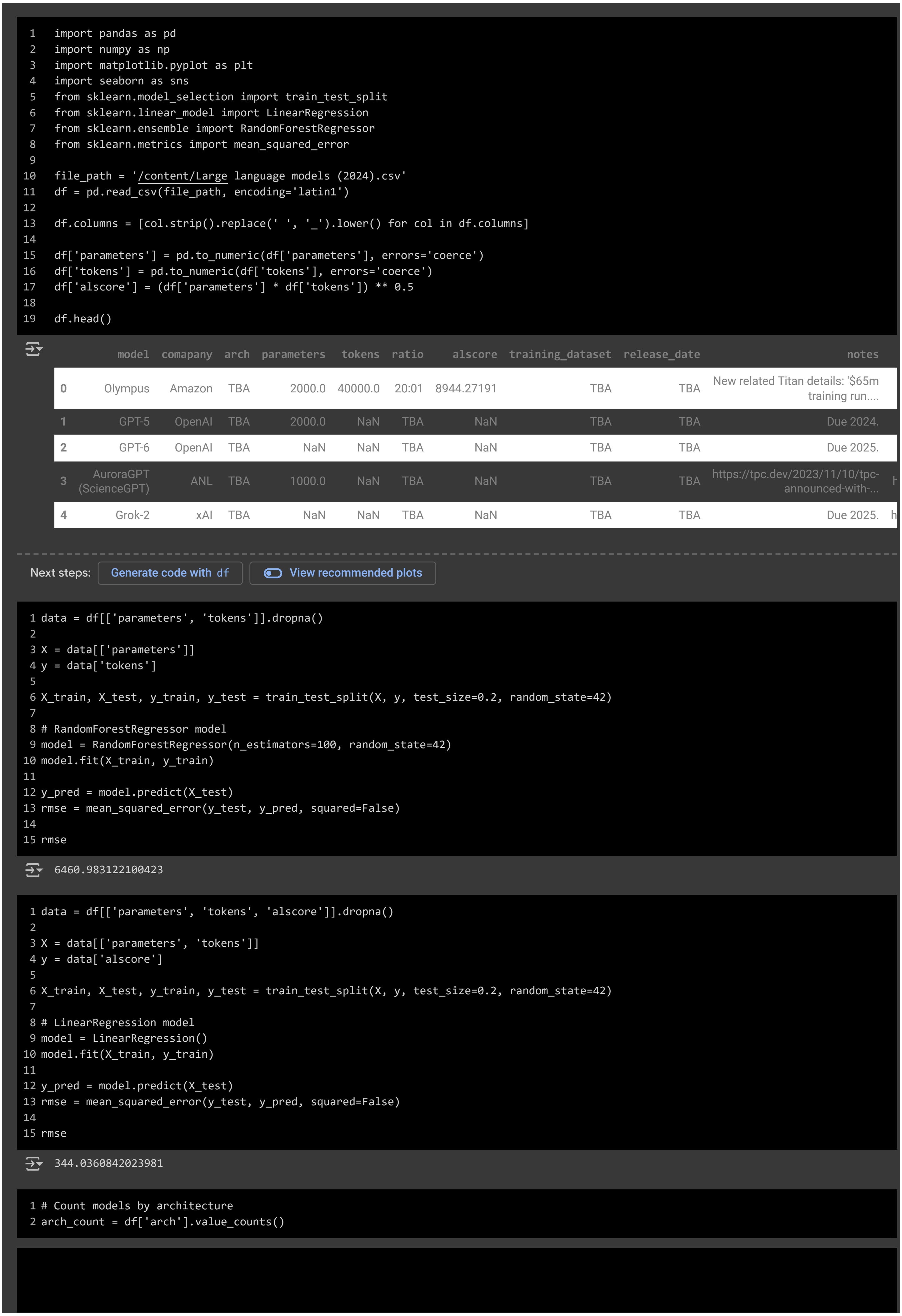


```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error

file_path = '/content/Large language models (2024).csv'
df = pd.read_csv(file_path, encoding='latin1')

df.columns = [col.strip().replace(' ', '_').lower() for col in df.columns]

df['parameters'] = pd.to_numeric(df['parameters'], errors='coerce')
df['tokens'] = pd.to_numeric(df['tokens'], errors='coerce')
df['alscore'] = (df['parameters'] * df['tokens']) ** 0.5

df.head()
```

	model	comapany	arch	parameters	tokens	ratio	alscore	training_dataset	release_date	notes
0	Olympus	Amazon	TBA	2000.0	40000.0	20:01	8944.27191	TBA	TBA	New related Titan details: '$65m training run....
1	GPT-5	OpenAI	TBA	2000.0	NaN	TBA	NaN	TBA	TBA	Due 2024.
2	GPT-6	OpenAI	TBA	NaN	NaN	TBA	NaN	TBA	TBA	Due 2025.
3	AuroraGPT (ScienceGPT)	ANL	TBA	1000.0	NaN	TBA	NaN	TBA	TBA	https://tpc.dev/2023/11/10/tpc-announced-with-...
4	Grok-2	xAI	TBA	NaN	NaN	TBA	NaN	TBA	TBA	Due 2025.

```
data = df[['parameters', 'tokens']].dropna()

X = data[['parameters']]
y = data['tokens']

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

# RandomForestRegressor model
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
rmse = mean_squared_error(y_test, y_pred, squared=False)

rmse
```

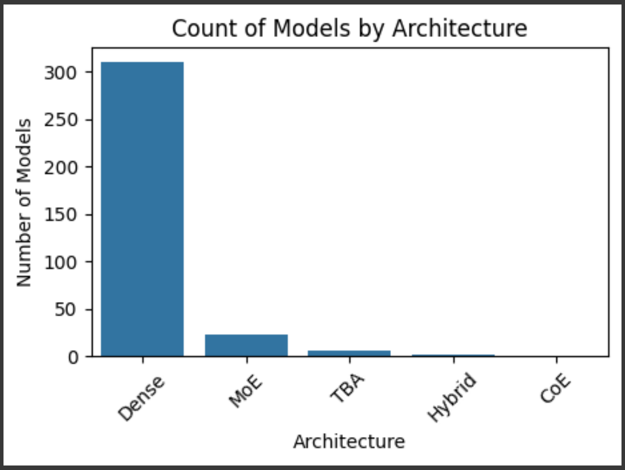
```
6460.983122100423
```

```
data = df[['parameters', 'tokens', 'alscore']].dropna()

X = data[['parameters', 'tokens']]
y = data['alscore']

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

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

y_pred = model.predict(X_test)
rmse = mean_squared_error(y_test, y_pred, squared=False)

rmse
```

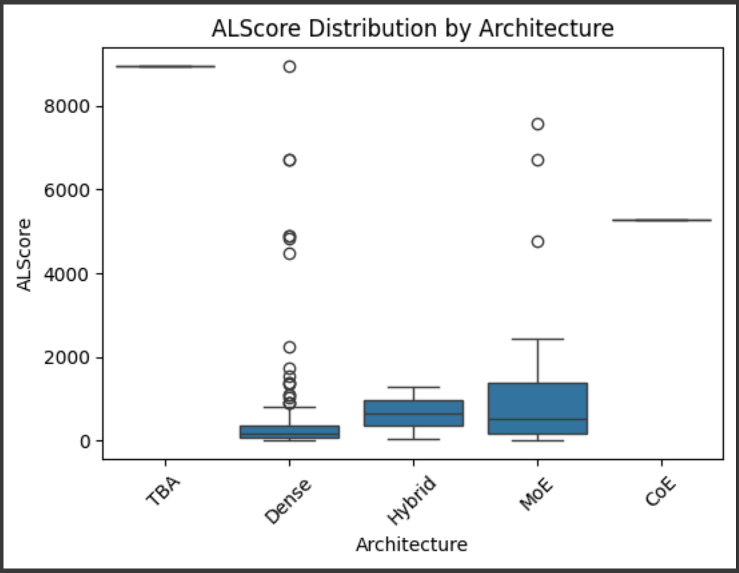
```
344.0360842023981
```

```
# Count models by architecture
arch_count = df['arch'].value_counts()
```

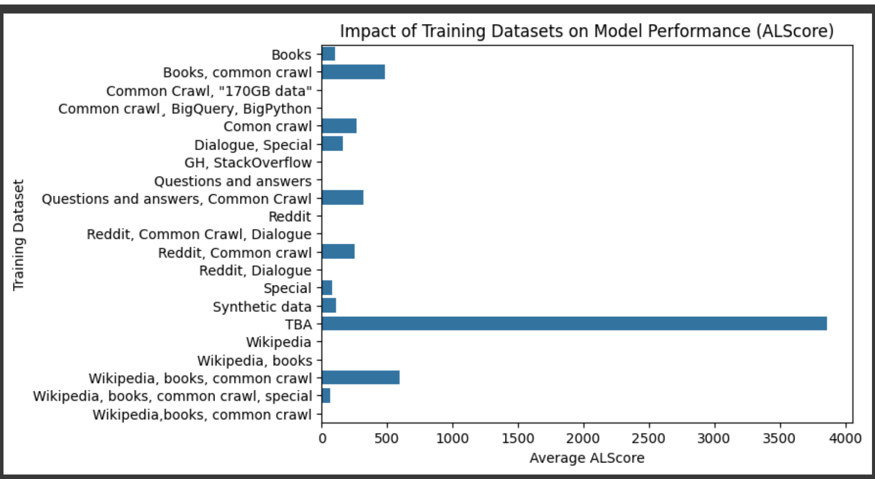
```python


# Plotting model count by architecture
plt.figure(figsize=(5, 3))
sns.barplot(x=arch_count.index, y=arch_count.values)
plt.title('Count of Models by Architecture')
plt.xlabel('Architecture')
plt.ylabel('Number of Models')
plt.xticks(rotation=45)
plt.show()
```



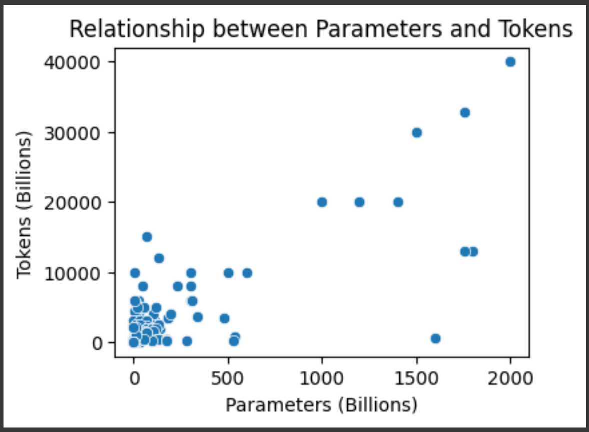
```python
# Boxplot of ALScore by architecture
plt.figure(figsize=(6, 4))
sns.boxplot(x='arch', y='alscore', data=df)
plt.title('ALScore Distribution by Architecture')
plt.xlabel('Architecture')
plt.ylabel('ALScore')
plt.xticks(rotation=45)
plt.show()
```



```python
# Analyzing the impact of training datasets
# Grouping by training dataset and calculate mean ALScore
dataset_impact = df.groupby('training_dataset')['alscore'].mean().reset_index()

# Plotting the impact of training datasets
plt.figure(figsize=(7, 5))
sns.barplot(x='alscore', y='training_dataset', data=dataset_impact)
plt.title('Impact of Training Datasets on Model Performance (ALScore)')
plt.xlabel('Average ALScore')
plt.ylabel('Training Dataset')
plt.show()
```



```python
# Scatter plot of parameters vs tokens
plt.figure(figsize=(4, 3))
sns.scatterplot(x='parameters', y='tokens', data=df)
plt.title('Relationship between Parameters and Tokens')
plt.xlabel('Parameters (Billions)')
plt.ylabel('Tokens (Billions)')
plt.show()
```


Relationship between Parameters and Tokens
Tokens (Billions)
Parameters (Billions)