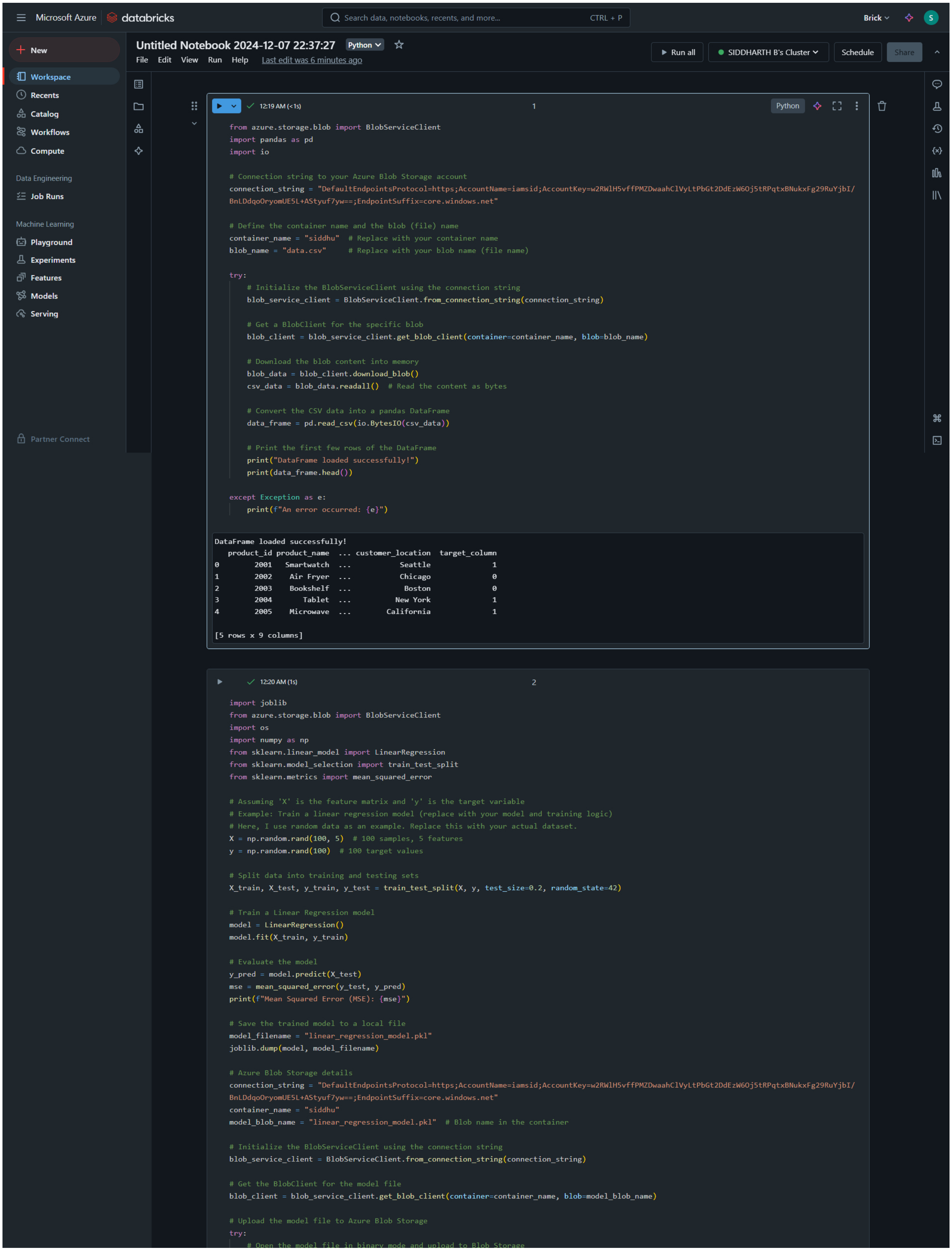
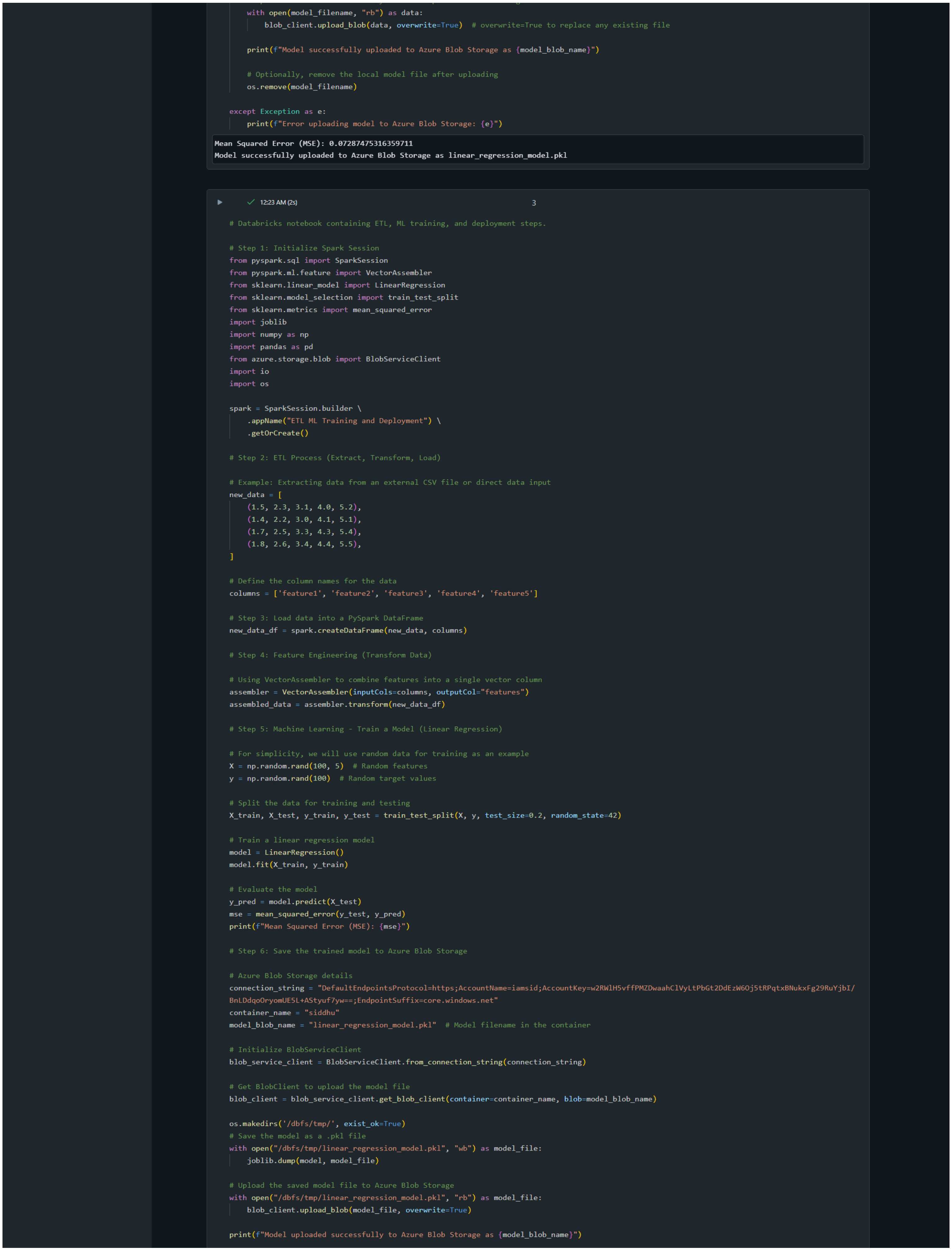


```python
from azure.storage.blob import BlobServiceClient
import pandas as pd
import io

# Connection string to your Azure Blob Storage account
connection_string = "DefaultEndpointsProtocol=https;AccountName=iamsid;AccountKey=w2RWlH5vffPMZDwaahClVyLtPbGt2DdEzW6Oj5tRPqtxBNukxFg29RuYjbI/
BnLDdqoOryomUE5L+AStyuf7yw==;EndpointSuffix=core.windows.net"

# Define the container name and the blob (file) name
container_name = "siddhu"  # Replace with your container name
blob_name = "data.csv"     # Replace with your blob name (file name)

try:
    # Initialize the BlobServiceClient using the connection string
    blob_service_client = BlobServiceClient.from_connection_string(connection_string)

    # Get a BlobClient for the specific blob
    blob_client = blob_service_client.get_blob_client(container=container_name, blob=blob_name)

    # Download the blob content into memory
    blob_data = blob_client.download_blob()
    csv_data = blob_data.readall()  # Read the content as bytes

    # Convert the CSV data into a pandas DataFrame
    data_frame = pd.read_csv(io.BytesIO(csv_data))

    # Print the first few rows of the DataFrame
    print("DataFrame loaded successfully!")
    print(data_frame.head())

except Exception as e:
    print(f"An error occurred: {e}")
```

```
DataFrame loaded successfully!
   product_id product_name  ... customer_location  target_column
0        2001   Smartwatch  ...           Seattle              1
1        2002    Air Fryer  ...           Chicago              0
2        2003    Bookshelf  ...            Boston              0
3        2004       Tablet  ...          New York              1
4        2005    Microwave  ...        California              1

[5 rows x 9 columns]
```

```python
import joblib
from azure.storage.blob import BlobServiceClient
import os
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Assuming 'X' is the feature matrix and 'y' is the target variable
# Example: Train a linear regression model (replace with your model and training logic)
# Here, I use random data as an example. Replace this with your actual dataset.
X = np.random.rand(100, 5)  # 100 samples, 5 features
y = np.random.rand(100)  # 100 target values

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Evaluate the model
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error (MSE): {mse}")

# Save the trained model to a local file
model_filename = "linear_regression_model.pkl"
joblib.dump(model, model_filename)

# Azure Blob Storage details
connection_string = "DefaultEndpointsProtocol=https;AccountName=iamsid;AccountKey=w2RWlH5vffPMZDwaahClVyLtPbGt2DdEzW6Oj5tRPqtxBNukxFg29RuYjbI/
BnLDdqoOryomUE5L+AStyuf7yw==;EndpointSuffix=core.windows.net"
container_name = "siddhu"
model_blob_name = "linear_regression_model.pkl"  # Blob name in the container

# Initialize the BlobServiceClient using the connection string
blob_service_client = BlobServiceClient.from_connection_string(connection_string)

# Get the BlobClient for the model file
blob_client = blob_service_client.get_blob_client(container=container_name, blob=model_blob_name)

# Upload the model file to Azure Blob Storage
try:
    # Open the model file in binary mode and upload to Blob Storage
```

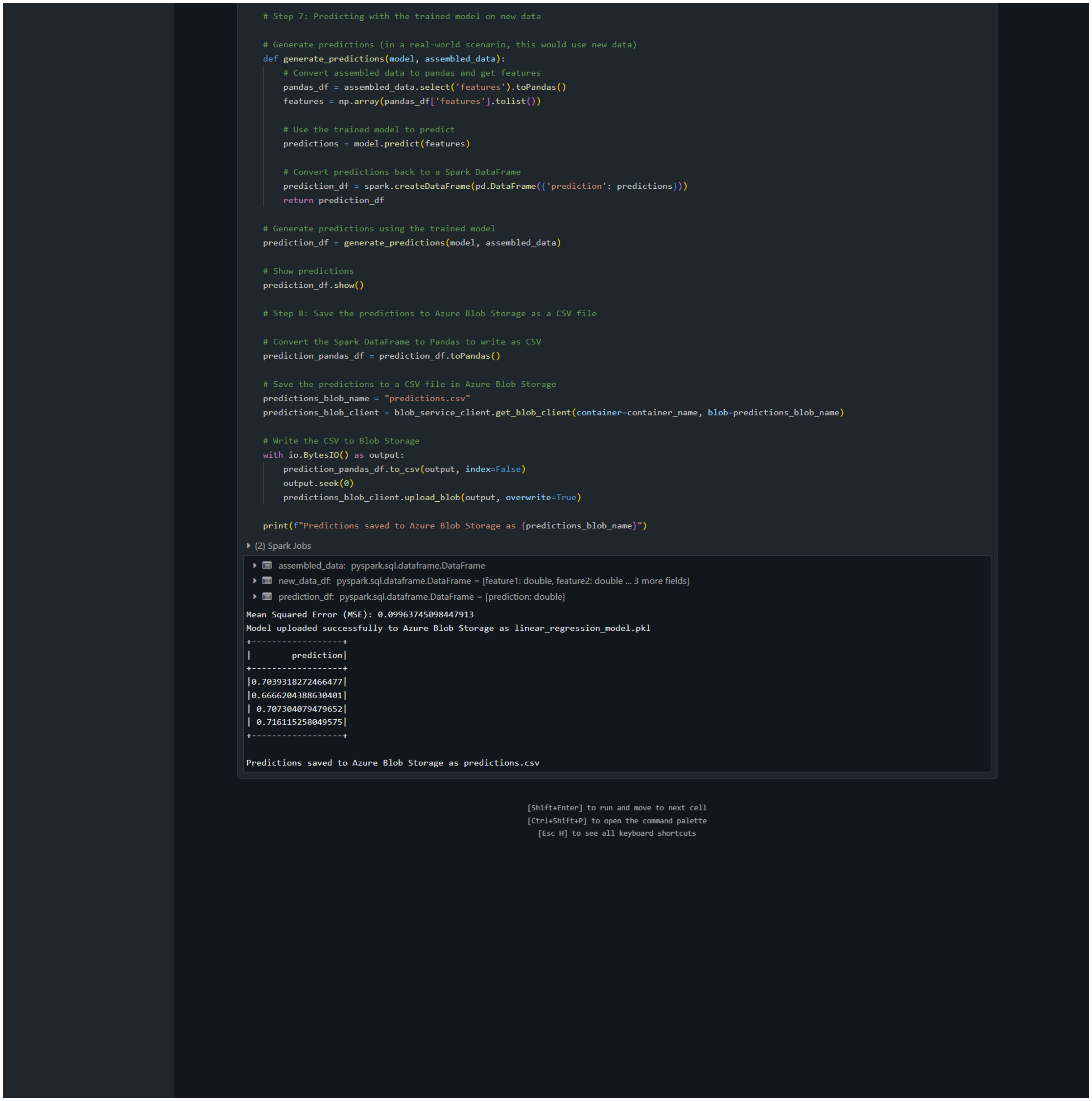


```
    with open(model_filename, "rb") as data:
        blob_client.upload_blob(data, overwrite=True)  # overwrite=True to replace any existing file

    print(f"Model successfully uploaded to Azure Blob Storage as {model_blob_name}")

    # Optionally, remove the local model file after uploading
    os.remove(model_filename)

except Exception as e:
    print(f"Error uploading model to Azure Blob Storage: {e}")
```

```
Mean Squared Error (MSE): 0.07287475316359711
Model successfully uploaded to Azure Blob Storage as linear_regression_model.pkl
```

✓ 12:23 AM (2s)   3

```
# Databricks notebook containing ETL, ML training, and deployment steps.

# Step 1: Initialize Spark Session
from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import joblib
import numpy as np
import pandas as pd
from azure.storage.blob import BlobServiceClient
import io
import os

spark = SparkSession.builder \
    .appName("ETL ML Training and Deployment") \
    .getOrCreate()

# Step 2: ETL Process (Extract, Transform, Load)

# Example: Extracting data from an external CSV file or direct data input
new_data = [
    (1.5, 2.3, 3.1, 4.0, 5.2),
    (1.4, 2.2, 3.0, 4.1, 5.1),
    (1.7, 2.5, 3.3, 4.3, 5.4),
    (1.8, 2.6, 3.4, 4.4, 5.5),
]

# Define the column names for the data
columns = ['feature1', 'feature2', 'feature3', 'feature4', 'feature5']

# Step 3: Load data into a PySpark DataFrame
new_data_df = spark.createDataFrame(new_data, columns)

# Step 4: Feature Engineering (Transform Data)

# Using VectorAssembler to combine features into a single vector column
assembler = VectorAssembler(inputCols=columns, outputCol="features")
assembled_data = assembler.transform(new_data_df)

# Step 5: Machine Learning - Train a Model (Linear Regression)

# For simplicity, we will use random data for training as an example
X = np.random.rand(100, 5)  # Random features
y = np.random.rand(100)  # Random target values

# Split the data for training and testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Evaluate the model
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error (MSE): {mse}")

# Step 6: Save the trained model to Azure Blob Storage

# Azure Blob Storage details
connection_string = "DefaultEndpointsProtocol=https;AccountName=iamsid;AccountKey=w2RWlH5vffPMZDwaahClVyLtPbGt2DdEzW6Oj5tRPqtxBNukxFg29RuYjbI/
BnLDdqoOryomUE5L+AStyuf7yw==;EndpointSuffix=core.windows.net"
container_name = "siddhu"
model_blob_name = "linear_regression_model.pkl"  # Model filename in the container

# Initialize BlobServiceClient
blob_service_client = BlobServiceClient.from_connection_string(connection_string)

# Get BlobClient to upload the model file
blob_client = blob_service_client.get_blob_client(container=container_name, blob=model_blob_name)

os.makedirs('/dbfs/tmp/', exist_ok=True)
# Save the model as a .pkl file
with open("/dbfs/tmp/linear_regression_model.pkl", "wb") as model_file:
    joblib.dump(model, model_file)

# Upload the saved model file to Azure Blob Storage
with open("/dbfs/tmp/linear_regression_model.pkl", "rb") as model_file:
    blob_client.upload_blob(model_file, overwrite=True)

print(f"Model uploaded successfully to Azure Blob Storage as {model_blob_name}")
```



```python
# Step 7: Predicting with the trained model on new data

# Generate predictions (in a real-world scenario, this would use new data)
def generate_predictions(model, assembled_data):
    # Convert assembled data to pandas and get features
    pandas_df = assembled_data.select('features').toPandas()
    features = np.array(pandas_df['features'].tolist())

    # Use the trained model to predict
    predictions = model.predict(features)

    # Convert predictions back to a Spark DataFrame
    prediction_df = spark.createDataFrame(pd.DataFrame({'prediction': predictions}))
    return prediction_df

# Generate predictions using the trained model
prediction_df = generate_predictions(model, assembled_data)

# Show predictions
prediction_df.show()

# Step 8: Save the predictions to Azure Blob Storage as a CSV file

# Convert the Spark DataFrame to Pandas to write as CSV
prediction_pandas_df = prediction_df.toPandas()

# Save the predictions to a CSV file in Azure Blob Storage
predictions_blob_name = "predictions.csv"
predictions_blob_client = blob_service_client.get_blob_client(container=container_name, blob=predictions_blob_name)

# Write the CSV to Blob Storage
with io.BytesIO() as output:
    prediction_pandas_df.to_csv(output, index=False)
    output.seek(0)
    predictions_blob_client.upload_blob(output, overwrite=True)

print(f"Predictions saved to Azure Blob Storage as {predictions_blob_name}")
```

▸ (2) Spark Jobs

▸ assembled_data: pyspark.sql.dataframe.DataFrame
▸ new_data_df: pyspark.sql.dataframe.DataFrame = [feature1: double, feature2: double ... 3 more fields]
▸ prediction_df: pyspark.sql.dataframe.DataFrame = [prediction: double]

```
Mean Squared Error (MSE): 0.09963745098447913
Model uploaded successfully to Azure Blob Storage as linear_regression_model.pkl
+------------------+
|        prediction|
+------------------+
|0.7039318272466477|
|0.6662043886304O1|
| 0.707304079479652|
| 0.716115258049575|
+------------------+


Predictions saved to Azure Blob Storage as predictions.csv
```

```
[Shift+Enter] to run and move to next cell
[Ctrl+Shift+P] to open the command palette
[Esc H] to see all keyboard shortcuts
```