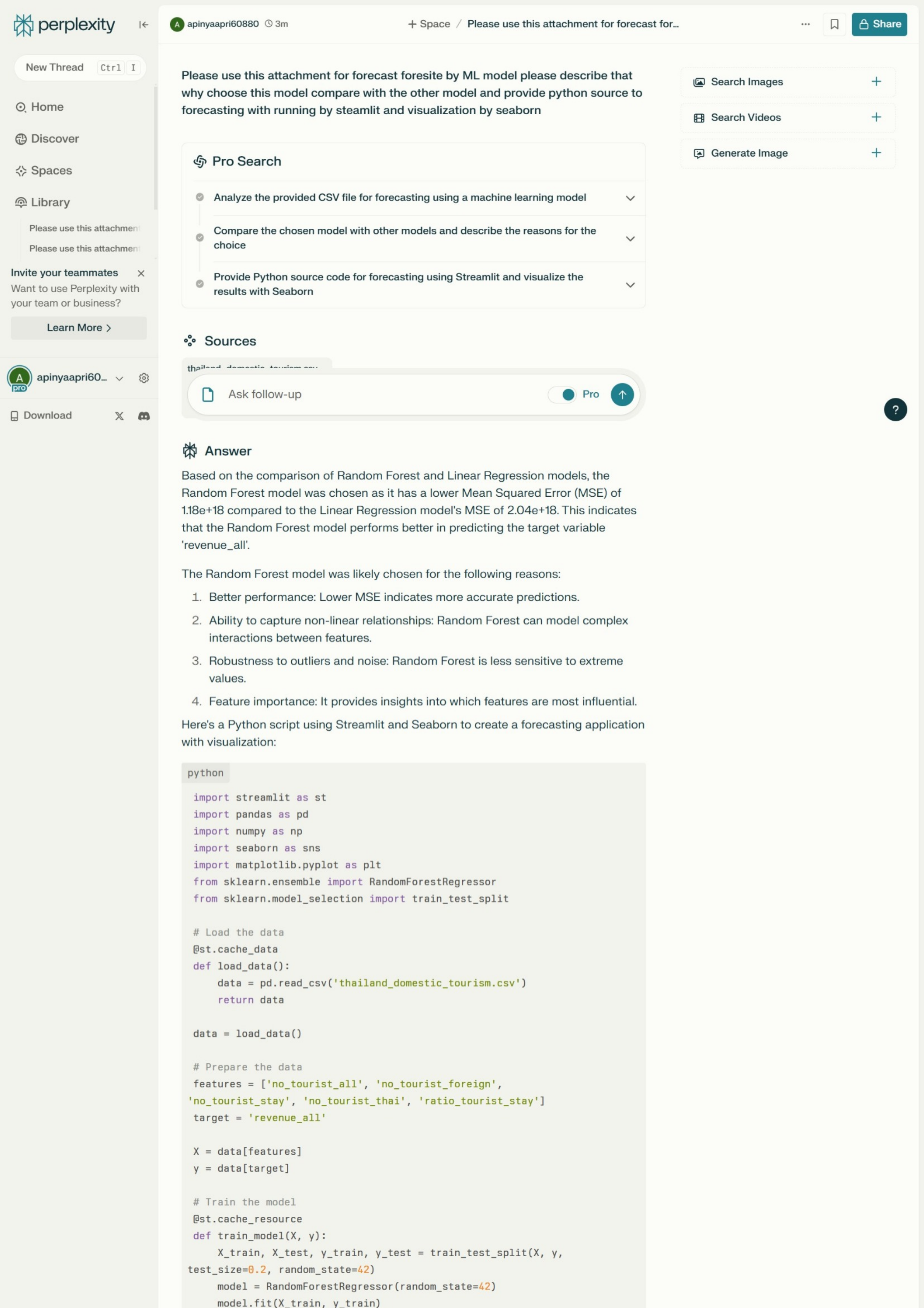

Please use this attachment for forecast foresite by ML model please describe that why choose this model compare with the other model and provide python source to forecasting with running by steamlit and visualization by seaborn

- Analyze the provided CSV file for forecasting using a machine learning model
- Compare the chosen model with other models and describe the reasons for the choice
- Provide Python source code for forecasting using Streamlit and visualize the results with Seaborn

Answer

Based on the comparison of Random Forest and Linear Regression models, the Random Forest model was chosen as it has a lower Mean Squared Error (MSE) of 1.18e+18 compared to the Linear Regression model's MSE of 2.04e+18. This indicates that the Random Forest model performs better in predicting the target variable 'revenue_all'.

The Random Forest model was likely chosen for the following reasons:

1. Better performance: Lower MSE indicates more accurate predictions.
2. Ability to capture non-linear relationships: Random Forest can model complex interactions between features.
3. Robustness to outliers and noise: Random Forest is less sensitive to extreme values.
4. Feature importance: It provides insights into which features are most influential.

Here's a Python script using Streamlit and Seaborn to create a forecasting application with visualization:

```python
import streamlit as st
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split

# Load the data
@st.cache_data
def load_data():
    data = pd.read_csv('thailand_domestic_tourism.csv')
    return data

data = load_data()

# Prepare the data
features = ['no_tourist_all', 'no_tourist_foreign',
'no_tourist_stay', 'no_tourist_thai', 'ratio_tourist_stay']
target = 'revenue_all'

X = data[features]
y = data[target]

# Train the model
@st.cache_resource
def train_model(X, y):
    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
    model = RandomForestRegressor(random_state=42)
    model.fit(X_train, y_train)
    return model

model = train_model(X, y)

# Streamlit app
st.title('Thailand Domestic Tourism Revenue Forecast')

st.sidebar.header('Input Parameters')
input_features = {}
for feature in features:
    input_features[feature] = st.sidebar.number_input(f'Enter {feature}', value=float(X[feature].mean()))

# Make prediction
input_df = pd.DataFrame([input_features])
prediction = model.predict(input_df)[0]

st.write(f'Predicted Revenue: {prediction:.2f}')

# Feature importance plot
st.subheader('Feature Importance')
feature_importance = pd.DataFrame({'feature': features, 'importance': model.feature_importances_})
feature_importance = feature_importance.sort_values('importance', ascending=False)

fig, ax = plt.subplots(figsize=(10, 6))
sns.barplot(x='importance', y='feature', data=feature_importance, ax=ax)
ax.set_title('Feature Importance')
st.pyplot(fig)

# Scatter plot of actual vs predicted values
st.subheader('Actual vs Predicted Revenue')
y_pred = model.predict(X)
fig, ax = plt.subplots(figsize=(10, 6))
sns.scatterplot(x=y, y=y_pred, ax=ax)
ax.set_xlabel('Actual Revenue')
ax.set_ylabel('Predicted Revenue')
ax.set_title('Actual vs Predicted Revenue')
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'r--', lw=2)
st.pyplot(fig)

# Distribution of prediction errors
st.subheader('Distribution of Prediction Errors')
errors = y - y_pred
fig, ax = plt.subplots(figsize=(10, 6))
sns.histplot(errors, kde=True, ax=ax)
ax.set_xlabel('Prediction Error')
ax.set_title('Distribution of Prediction Errors')
st.pyplot(fig)
```

To run this Streamlit app:

1. Save the script as `` `app.py` ``
2. Ensure you have the required libraries installed: `` `streamlit` ``, `` `pandas` ``, `` `numpy` ``, `` `seaborn` ``, `` `matplotlib` ``, `` `scikit-learn` ``
3. Place the `` `thailand_domestic_tourism.csv` `` file in the same directory as the script
4. Run the command: `` `streamlit run app.py` ``

This app allows users to input feature values, displays the predicted revenue, shows feature importance, visualizes actual vs predicted values, and presents the distribution of prediction errors.

Share Rewrite

Related

What are the key features of the ML model used for forecasting +

How does the chosen ML model compare to other models in terms of accuracy +

Can you provide a step-by-step guide for implementing the ML model in Python +

What are the advantages of using Streamlit for deploying the forecasting model +

How can Seaborn be utilized for visualizing the forecasting results +