```
# Customer Churn Analysis for a Subscription Business

# Goal: Identify Customers likely to churn and understand key churn factors
# Business Questions:
# Who are the customers likely to churn?
# What are the key influencers of churn?
# What actions can reduce churn?


# Load Data

import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Dataset/Telco-Customer-Churn.csv')


# Handle Missing Values

df.isnull().sum()
df = df.dropna()
df
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No
...	...	...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes

7043 rows × 21 columns

```
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df.dropna()
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No
...	...	...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes

7032 rows × 21 columns

```
df['Churn_Yes'] = df['Churn'].map({'Yes':1, 'No':0})
df
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No
...	...	...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes

7043 rows × 22 columns

```
df['TenureGroup'] = pd.cut(df['tenure'], bins=[0, 12, 24, 48, 60, 72],
                           labels=['0-12','12-24','24-48','48-60','60-72'])
df
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No
...	...	...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes

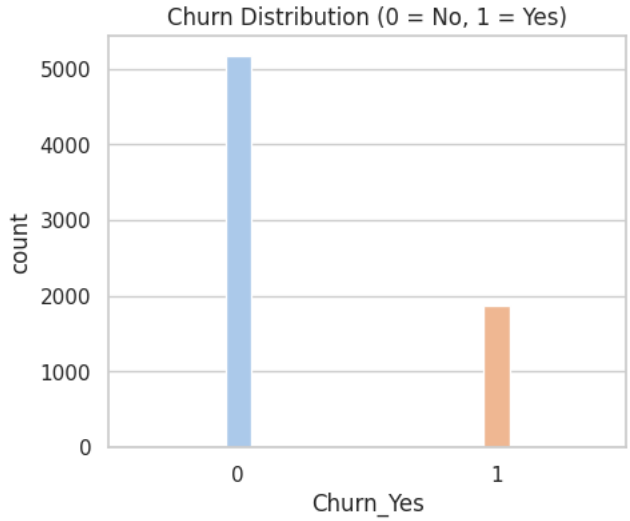
7043 rows × 23 columns

```
df.drop(columns=['customerID'], inplace=True)

df.head()
```

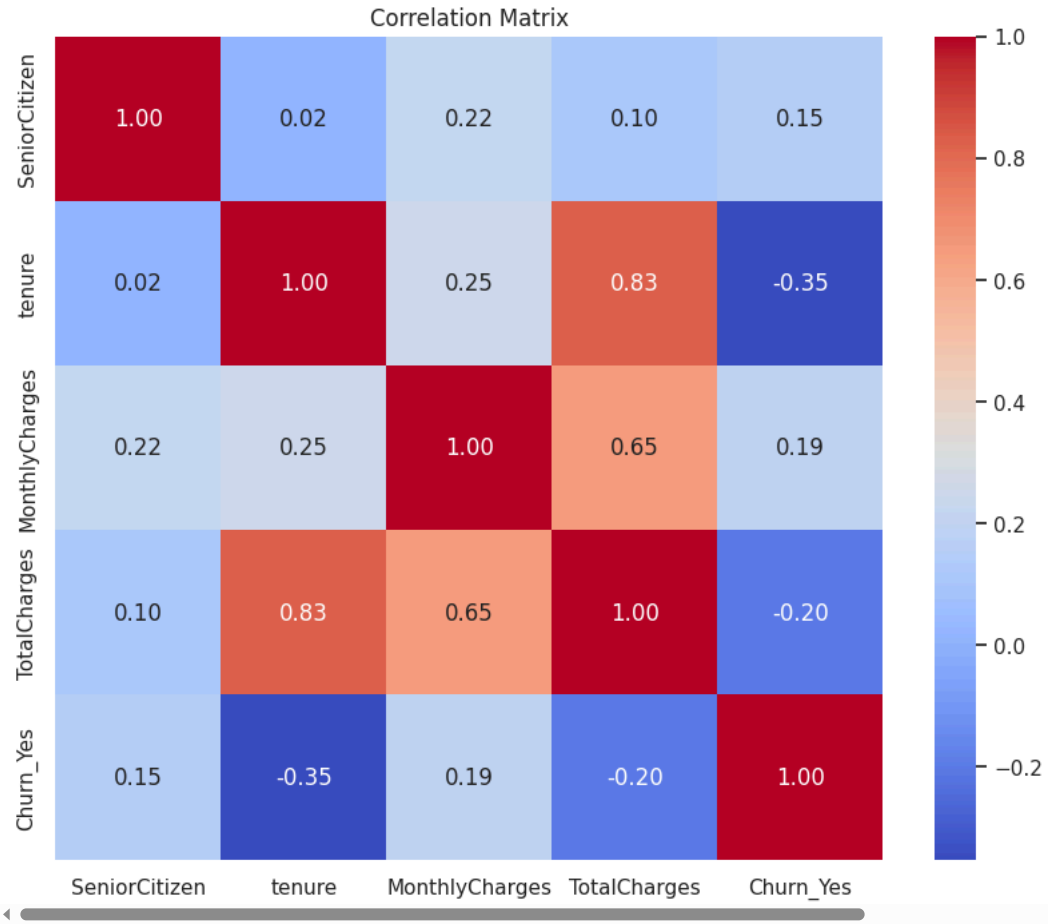
	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	.
0	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes	
1	Male	0	No	No	34	Yes	No	DSL	Yes	No	
2	Male	0	No	No	2	Yes	No	DSL	Yes	Yes	
3	Male	0	No	No	45	No	No phone service	DSL	Yes	No	
4	Female	0	No	No	2	Yes	No	Fiber optic	No	No	

5 rows × 22 columns

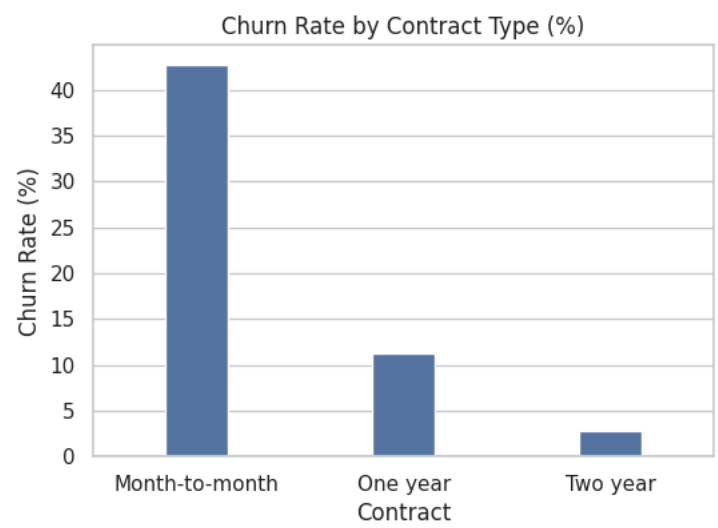
```
# Exploratory Data Analysis


import seaborn as sns
import matplotlib.pyplot as plt

# Set theme
sns.set(style='whitegrid')


# Churn Distribution

plt.figure(figsize=(5,4))
sns.countplot(x='Churn_Yes', data=df, palette='pastel', width=0.1)
plt.title('Churn Distribution (0 = No, 1 = Yes)')
plt.show()
```



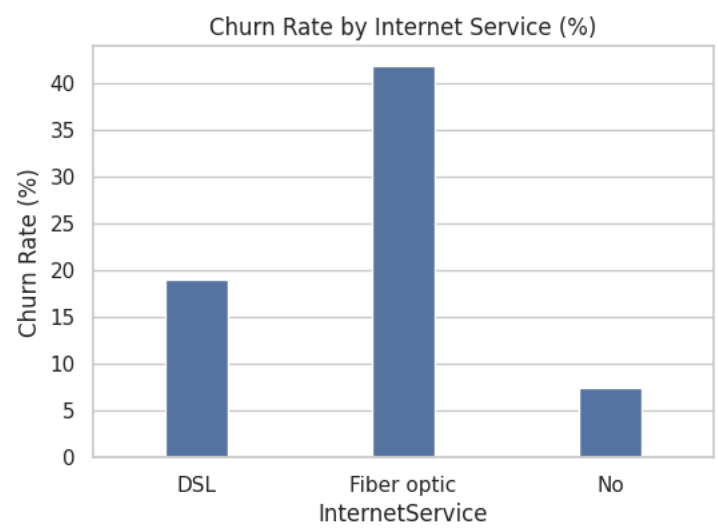

```
# 2. Correlation Matrix (numerical features)

plt.figure(figsize=(10,8))
corr = df.select_dtypes(include=['float64', 'int64']).corr()
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Matrix")
plt.show()
```



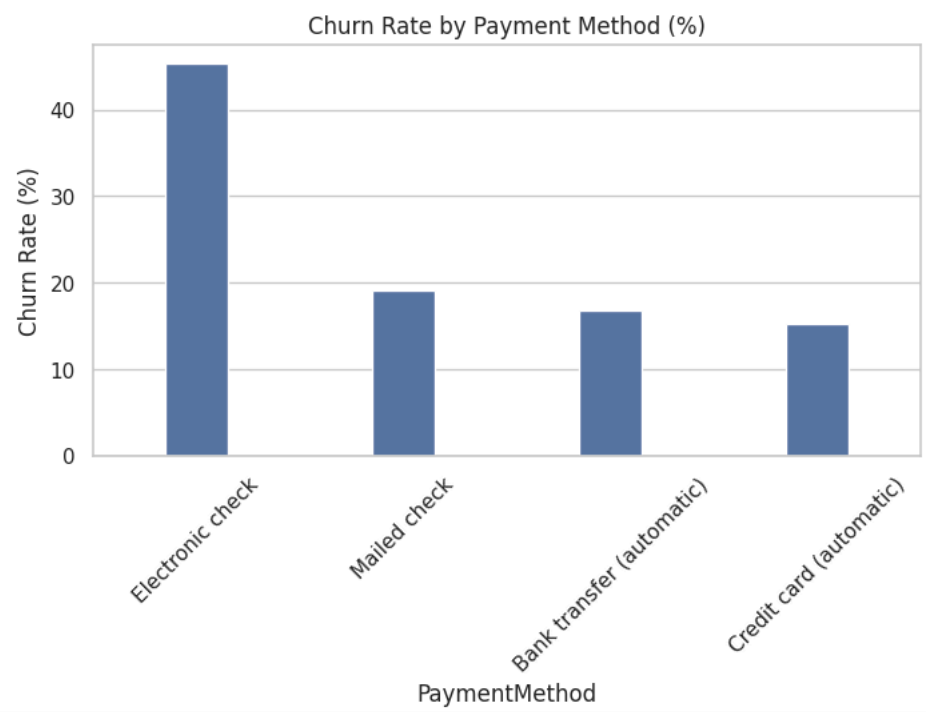

```
# 3. Churn by Contract Type

plt.figure(figsize=(6,4))
sns.barplot(data=df, x='Contract', y='Churn_Yes', width=0.3, estimator=lambda x: sum(x) / len(x) * 100, ci=None)
plt.title("Churn Rate by Contract Type (%)")
plt.ylabel("Churn Rate (%)")
plt.show()
```



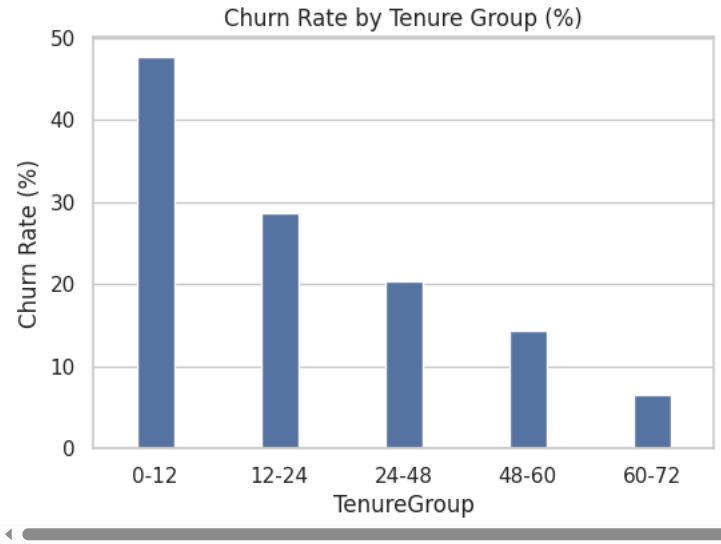

```
# 4. Churn by Internet Service

plt.figure(figsize=(6,4))
sns.barplot(data=df, x='InternetService', y='Churn_Yes', width=0.3, estimator=lambda x: sum(x) / len(x) * 100, ci=None)
plt.title("Churn Rate by Internet Service (%)")
plt.ylabel("Churn Rate (%)")
plt.show()
```




```
# 5. Churn by Payment Method

plt.figure(figsize=(8,4))
sns.barplot(data=df, x='PaymentMethod', y='Churn_Yes', width=0.3, estimator=lambda x: sum(x) / len(x) * 100, ci=None)
plt.title("Churn Rate by Payment Method (%)")
plt.ylabel("Churn Rate (%)")
plt.xticks(rotation=45)
plt.show()
```




```
# 6. Churn by Tenure Group

plt.figure(figsize=(6,4))
sns.barplot(data=df, x='TenureGroup', y='Churn_Yes', width=0.3, estimator=lambda x: sum(x) / len(x) * 100, ci=None)
plt.title("Churn Rate by Tenure Group (%)")
plt.ylabel("Churn Rate (%)")
plt.show()
```




```
df
```

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup
0	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes
1	Male	0	No	No	34	Yes	No	DSL	Yes	No
2	Male	0	No	No	2	Yes	No	DSL	Yes	Yes
3	Male	0	No	No	45	No	No phone service	DSL	Yes	No
4	Female	0	No	No	2	Yes	No	Fiber optic	No	No
...	...	...	...	...	...	...	...	...	...	...
7038	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	No
7039	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	Yes
7040	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	No
7041	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	No
7042	Male	0	No	No	66	Yes	No	Fiber optic	Yes	No

7043 rows × 22 columns

```
# Data Modeling


# Import Libraries

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
import pandas as pd
import matplotlib.pyplot as plt


# Prepare the data
# Drop non-predictive and target columns

X = df.drop(['Churn', 'Churn_Yes'], axis=1)


# Convert categorical variables to dummy/one-hot encoding

X = pd.get_dummies(X, drop_first=True)


# Target variable

y = df['Churn_Yes']


# Step 3: Train-test split

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


# Train the model

model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

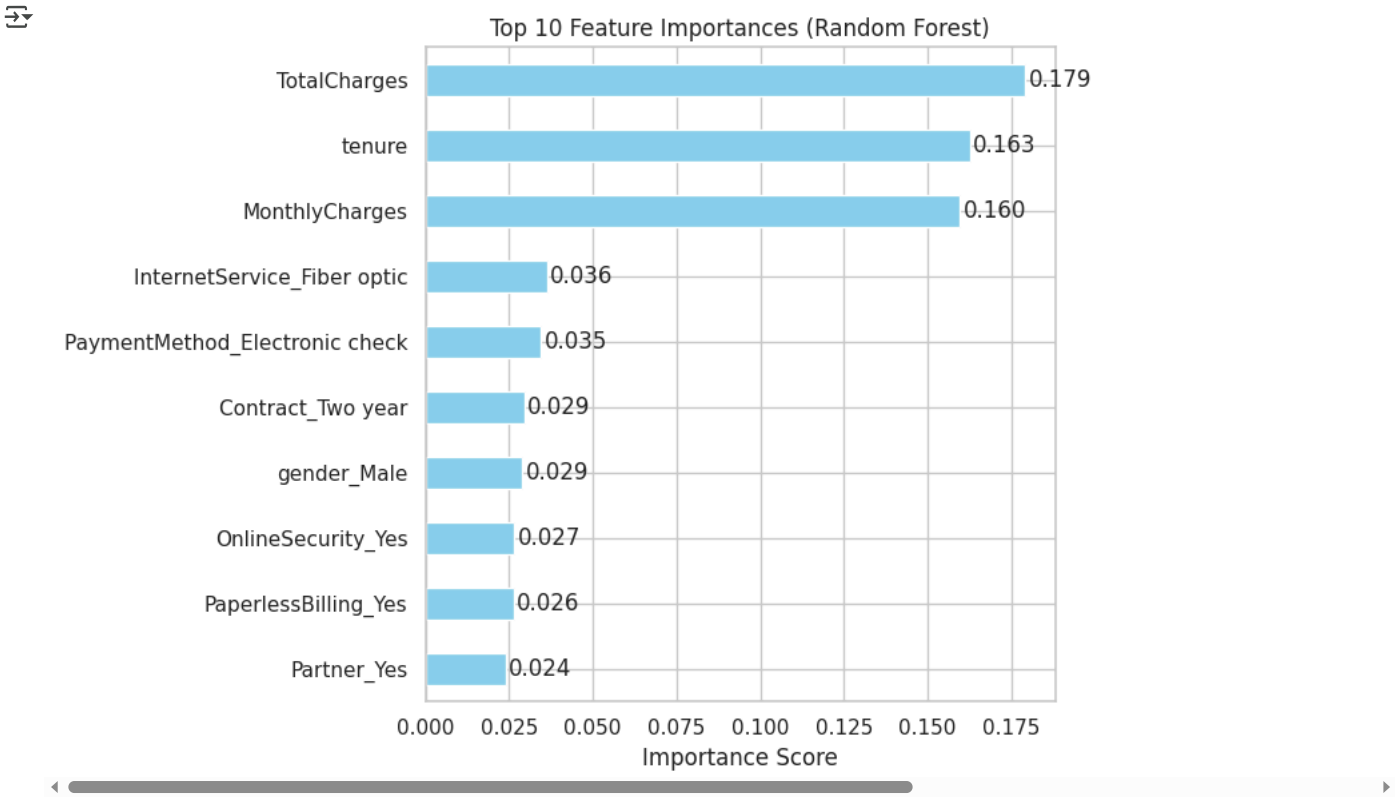
```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
# Predictions
y_pred = model.predict(X_test)


# Evaluation
print("Classification Report:\n", classification_report(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support
```

```
              0       0.82      0.92      0.87      1036
              1       0.66      0.46      0.54       373

       accuracy                           0.79      1409
      macro avg       0.74      0.69      0.70      1409
   weighted avg       0.78      0.79      0.78      1409

   Confusion Matrix:
    [[949  87]
    [202 171]]
```

```
# Feature Importance Plot


# Get top 10 important features

importances = pd.Series(model.feature_importances_, index=X.columns)
top_features = importances.sort_values(ascending=False).head(10)


# Plot with labels

plt.figure(figsize=(8,6))
ax = top_features.plot(kind='barh', color='skyblue')
plt.title('Top 10 Feature Importances (Random Forest)')
plt.xlabel('Importance Score')
plt.gca().invert_yaxis()

# Add labels

for i, (value, name) in enumerate(zip(top_features, top_features.index)):
    plt.text(value + 0.001, i, f'{value:.3f}', va='center')

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




```
from sklearn.model_selection import GridSearchCV
```