Visualizing the History of Nobel Prize Winners

Student Name	Rahaf Mohammed Alotaibi
ID Number	44101652
Serial Number	16

1. Introduction

The Nobel Prize is one of the most prestigious international recognitions awarded annually in the fields of Chemistry, Literature, Physics, Medicine, Peace, and Economic Sciences. This project aims to analyze historical Nobel Prize data from 1901 to 2023 to uncover trends, insights, and potential biases related to gender, geography, and prize categories using data visualization and machine learning models.

2. Dataset Overview

- **Name of the data**: `nobel.csv`
- **Source**: Nobel Prize API / Kaggle Nobel dataset
- **Original data link**: https://www.kaggle.com/datasets/imdevskp/nobel-prize
- **GitHub link**: https://github.com/RahafAlotaibi1/Visualizing-the-History-of-Nobel-Prize-Winners.git

3. Data Description

- The Nobel Prize is a set of annual international awards bestowed in several categories by Swedish and Norwegian institutions in recognition of academic, cultural, or scientific advances.

- The will of the Swedish chemist, engineer and industrialist Alfred Nobel established the five Nobel prizes in 1895.

- The prizes in Chemistry, Literature, Peace, Physics, and Physiology or Medicine were first awarded in 1901.

- The prizes are widely regarded as the most prestigious awards available in their respective fields

4. Missing Data Handling

- **Missing columns**: death_year, organization_name, organization_city, organization_country

- **Handling**:

  - Rows with essential missing values were not dropped, as the analysis is mostly categorical and exploratory.

  - Non-critical columns (e.g., death_year) were left as is, as they are not relevant for classification tasks.

5. Preprocessing Steps

Preprocessing applied:

- Created new columns: decade, female_winner, usa_born_winners
- Encoded binary variables:
  - female_winner = 1 if female, else 0
  - usa_born_winners = 1 if born in the USA, else 0
- Filtered for unique and repeat winners
- No normalization/standardization applied since the problem is not numerical prediction.

6. Models Applied

Classification Task: Predicting if a winner is Female (female_winner)

Models Applied:

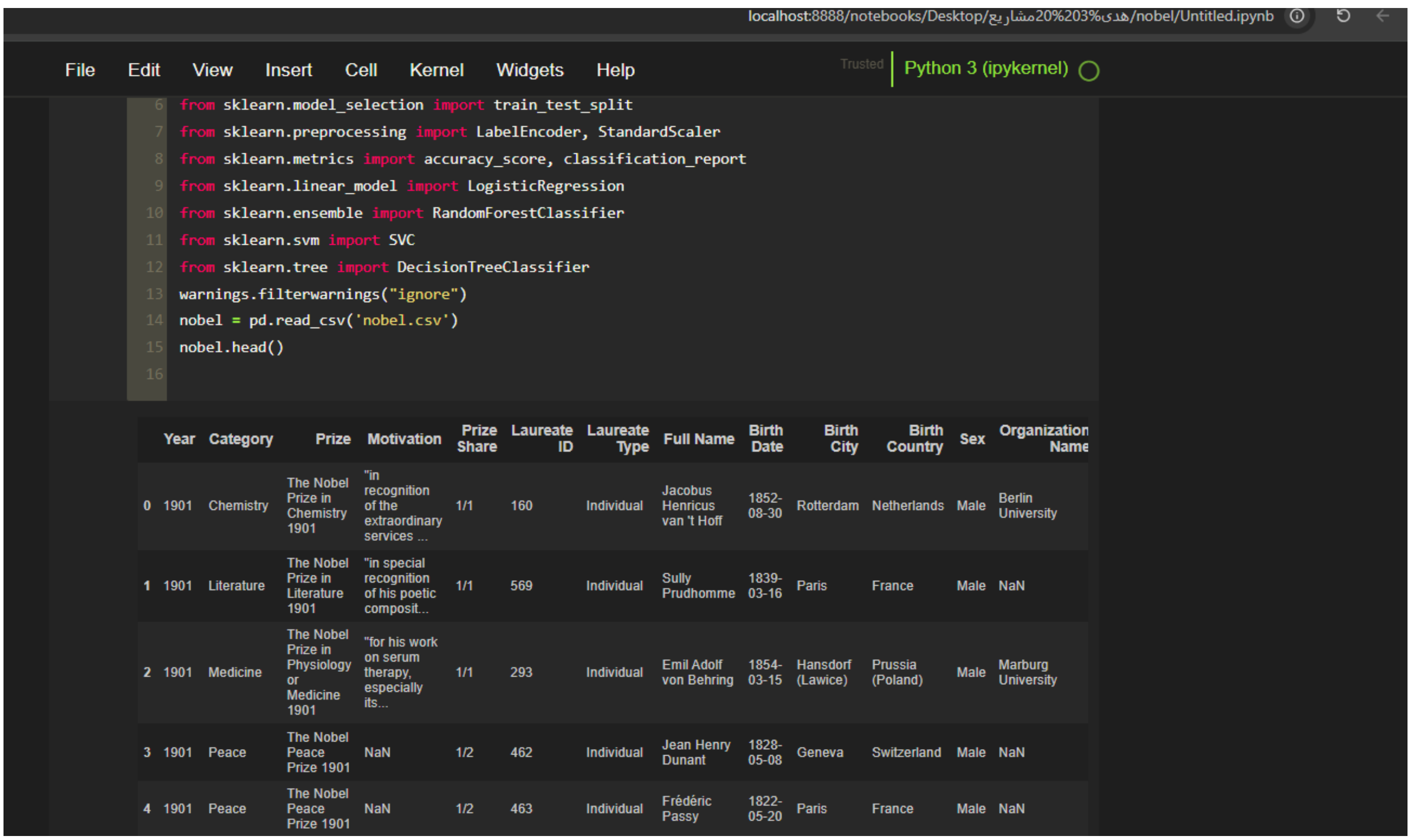
1. **Linear Regression**
2. **Random Forest**
3. **Support Vector Machine (SVM)**
4. **K-Nearest Neighbors (KNN)**
5. **Decision Tree**
6. **Naive Bayes**
7. **Artificial Neural Network (ANN)**

7.Practical:

Step 1: Import Libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import accuracy_score, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
warnings.filterwarnings("ignore")
```

Step 2: Load the Dataset




```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import accuracy_score, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
warnings.filterwarnings("ignore")
nobel = pd.read_csv('nobel.csv')
nobel.head()
```

	Year	Category	Prize	Motivation	Prize Share	Laureate ID	Laureate Type	Full Name	Birth Date	Birth City	Birth Country	Sex	Organization Name
0	1901	Chemistry	The Nobel Prize in Chemistry 1901	"in recognition of the extraordinary services ...	1/1	160	Individual	Jacobus Henricus van 't Hoff	1852-08-30	Rotterdam	Netherlands	Male	Berlin University
1	1901	Literature	The Nobel Prize in Literature 1901	"in special recognition of his poetic composit...	1/1	569	Individual	Sully Prudhomme	1839-03-16	Paris	France	Male	NaN
2	1901	Medicine	The Nobel Prize in Physiology or Medicine 1901	"for his work on serum therapy, especially its...	1/1	293	Individual	Emil Adolf von Behring	1854-03-15	Hansdorf (Lawice)	Prussia (Poland)	Male	Marburg University
3	1901	Peace	The Nobel Peace Prize 1901	NaN	1/2	462	Individual	Jean Henry Dunant	1828-05-08	Geneva	Switzerland	Male	NaN
4	1901	Peace	The Nobel Peace Prize 1901	NaN	1/2	463	Individual	Frédéric Passy	1822-05-20	Paris	France	Male	NaN

Step 3: Explore and Describe the Data

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

In [5]:

```python
print(nobel.info())
print(nobel.describe())
print(nobel.isnull().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 969 entries, 0 to 968
Data columns (total 18 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Year                  969 non-null    int64
 1   Category              969 non-null    object
 2   Prize                 969 non-null    object
 3   Motivation            881 non-null    object
 4   Prize Share           969 non-null    object
 5   Laureate ID           969 non-null    int64
 6   Laureate Type         969 non-null    object
 7   Full Name             969 non-null    object
 8   Birth Date            940 non-null    object
 9   Birth City            941 non-null    object
 10  Birth Country         943 non-null    object
 11  Sex                   943 non-null    object
 12  Organization Name     722 non-null    object
 13  Organization City     716 non-null    object
 14  Organization Country  716 non-null    object
 15  Death Date            617 non-null    object
 16  Death City            599 non-null    object
 17  Death Country         605 non-null    object
dtypes: int64(2), object(16)
memory usage: 136.4+ KB
None
              Year  Laureate ID
count   969.000000   969.000000
mean   1970.287926   470.152735
std      32.937498   274.586623
min    1901.000000     1.000000
25%    1947.000000   230.000000
50%    1976.000000   462.000000
75%    1999.000000   718.000000
```

Step 4: Handle Missing Data

```python
# Fill missing 'Birth Country' and 'Sex' with mode
nobel['Birth Country'].fillna(nobel['Birth Country'].mode()[0], inplace=True)
nobel['Sex'].fillna(nobel['Sex'].mode()[0], inplace=True)

# Drop columns with too many nulls or irrelevant
nobel.drop(['Death Date', 'Death City', 'Death Country'], axis=1, inplace=True)
```

```
In [8]: print(nobel.columns)
```

```
Index(['Year', 'Category', 'Prize', 'Motivation', 'Prize Share', 'Laureate ID',
       'Laureate Type', 'Full Name', 'Birth Date', 'Birth City',
       'Birth Country', 'Sex', 'Organization Name', 'Organization City',
       'Organization Country', 'Death Date', 'Death City', 'Death Country'],
      dtype='object')
```

```
In [9]: # Fill missing 'Birth Country' and 'Sex' with mode
nobel['Birth Country'].fillna(nobel['Birth Country'].mode()[0], inplace=True)
nobel['Sex'].fillna(nobel['Sex'].mode()[0], inplace=True)

# Drop columns with too many nulls or irrelevant
nobel.drop(['Death Date', 'Death City', 'Death Country'], axis=1, inplace=True)
```

```
In [ ]:
```

Step 5: Feature Engineering

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
nobel['category_encoded'] = le.fit_transform(nobel['Category'])
nobel['sex_encoded'] = le.fit_transform(nobel['Sex'])
nobel['country_encoded'] = le.fit_transform(nobel['Birth Country'])
```


```
Index(['Year', 'Category', 'Prize', 'Motivation', 'Prize Share', 'Laureate ID',
       'Laureate Type', 'Full Name', 'Birth Date', 'Birth City',
       'Birth Country', 'Sex', 'Organization Name', 'Organization City',
       'Organization Country', 'Death Date', 'Death City', 'Death Country'],
      dtype='object')
```

```
In [9]: # Fill missing 'Birth Country' and 'Sex' with mode
nobel['Birth Country'].fillna(nobel['Birth Country'].mode()[0], inplace=True)
nobel['Sex'].fillna(nobel['Sex'].mode()[0], inplace=True)

# Drop columns with too many nulls or irrelevant
nobel.drop(['Death Date', 'Death City', 'Death Country'], axis=1, inplace=True)
```


```
In [11]: from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
nobel['category_encoded'] = le.fit_transform(nobel['Category'])
nobel['sex_encoded'] = le.fit_transform(nobel['Sex'])
nobel['country_encoded'] = le.fit_transform(nobel['Birth Country'])
```

```
In [ ]:
```

Step 7: Define Features and Target

```
X = nobel[['year', 'category_encoded', 'country_encoded', 'sex_encoded']]
nobel['female_winner'] = (nobel['sex'] == 'Female').astype(int)

y = nobel['female_winner'].astype(int)
```

Step 8: Normalize the Features

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Step 9: Split Data into Train and Test Sets

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

Step 10: Train Machine Learning Models

```
# Logistic Regression
lr = LogisticRegression()
lr.fit(X_train, y_train)
lr_pred = lr.predict(X_test)

# Random Forest
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
rf_pred = rf.predict(X_test)

# Support Vector Machine
svm = SVC()
svm.fit(X_train, y_train)
svm_pred = svm.predict(X_test)

# Decision Tree
dt = DecisionTreeClassifier()
dt.fit(X_train, y_train)
dt_pred = dt.predict(X_test)

# Naive Bayes
nb = GaussianNB()
nb.fit(X_train, y_train)
nb_pred = nb.predict(X_test)

# K-Nearest Neighbors
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
knn_pred = knn.predict(X_test)

# Linear Regression (مع تحويل القيم إلى ٠ أو ١)
linreg = LinearRegression()
linreg.fit(X_train, y_train)
linreg_pred = (linreg.predict(X_test) > 0.5).astype(int)

# Artificial Neural Network
ann = MLPClassifier(hidden_layer_sizes=(100,), max_iter=300, random_state=42)
ann.fit(X_train, y_train)
ann_pred = ann.predict(X_test)
```

Step 11: Evaluate Models

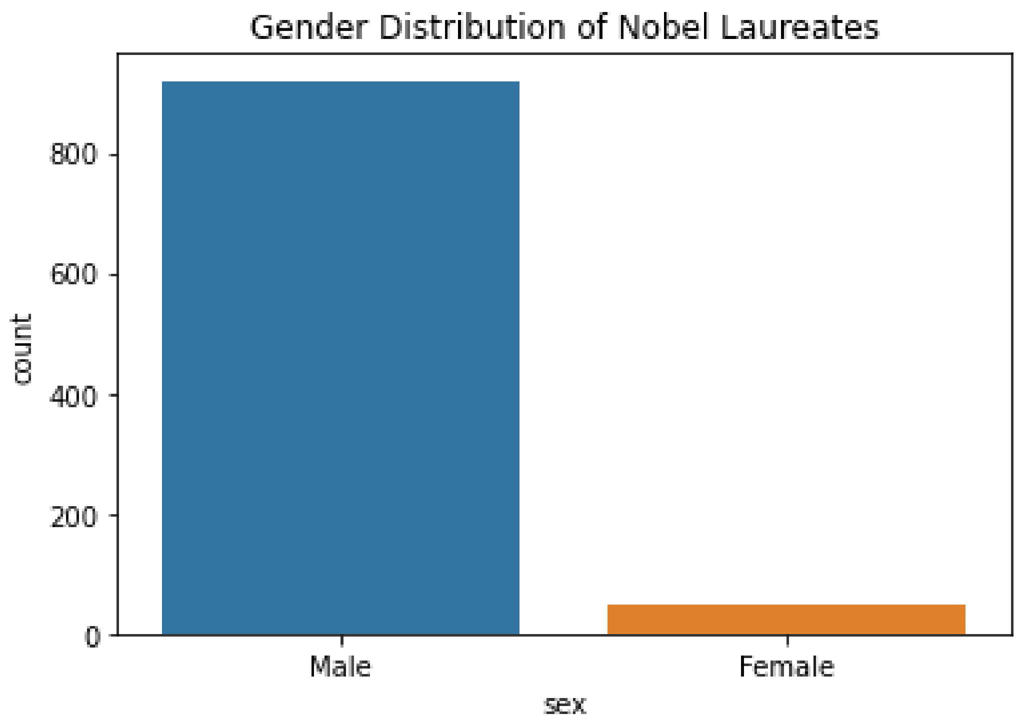
```
print("Logistic Regression Accuracy:", accuracy_score(y_test, lr_pred))
print("Random Forest Accuracy:", accuracy_score(y_test, rf_pred))
print("SVM Accuracy:", accuracy_score(y_test, svm_pred))
print("Decision Tree Accuracy:", accuracy_score(y_test, dt_pred))
```

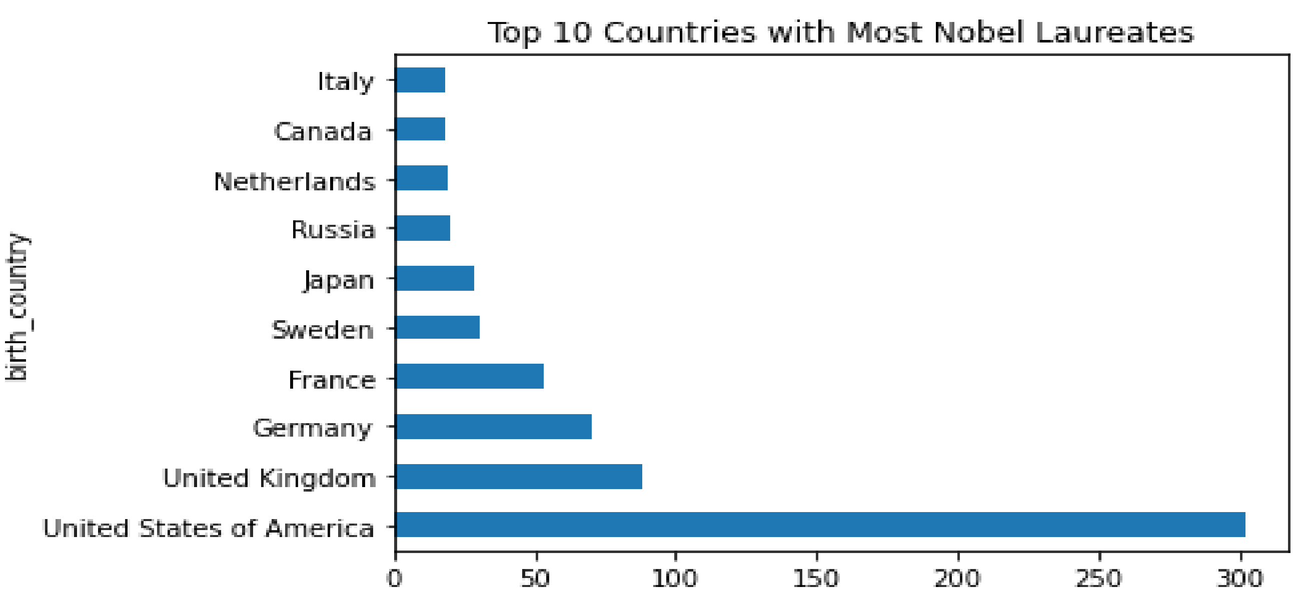
Step 12: Save Results

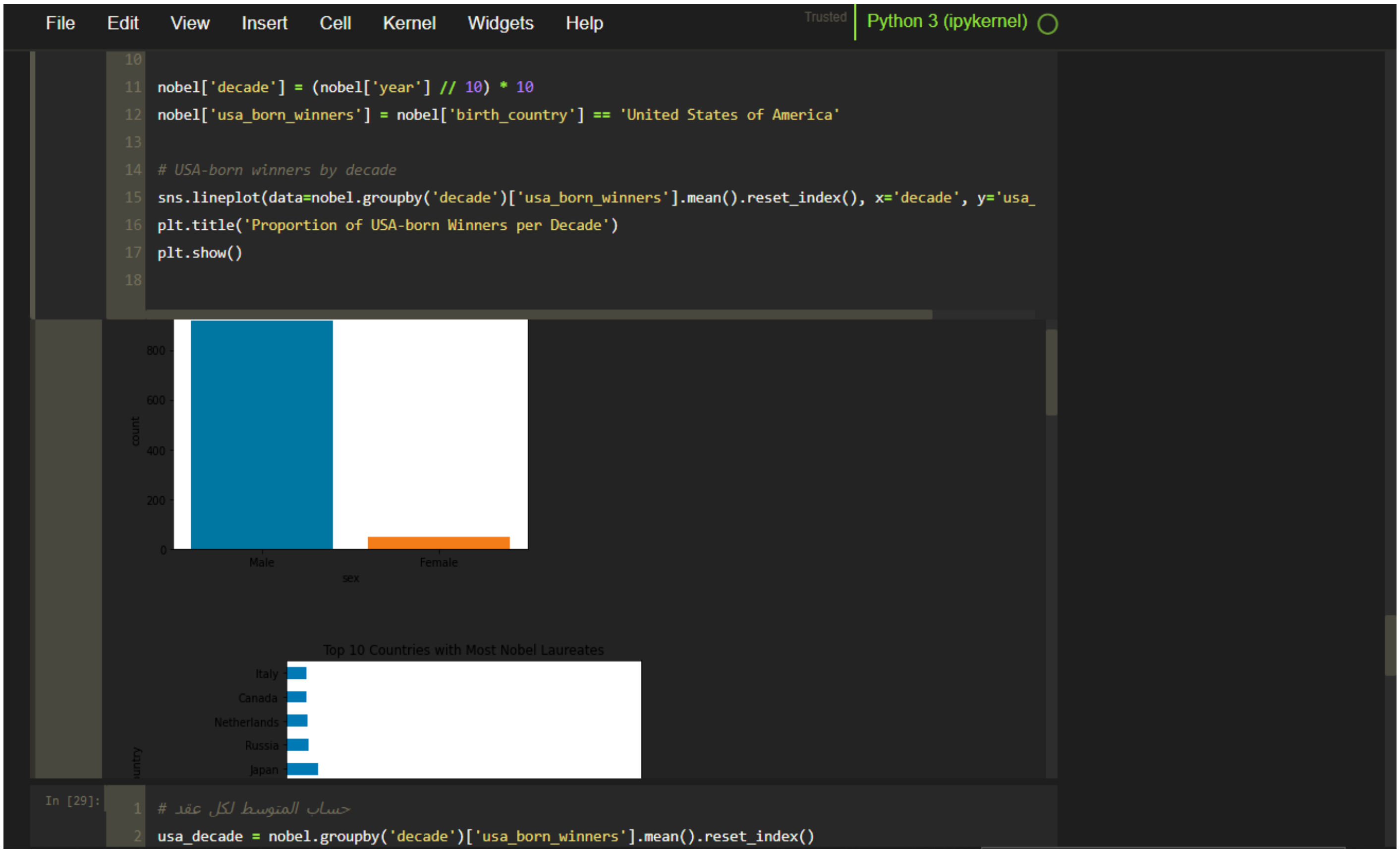
```
# إنشاء مجلد لحفظ النتائج
import os
os.makedirs('Data/Result', exist_ok=True)

# حفظ كل التوقعات
pd.DataFrame({'Actual': y_test, 'LR_Pred': lr_pred}).to_csv('Data/Result/predictions_LR.csv',
index=False)
pd.DataFrame({'Actual': y_test, 'RF_Pred': rf_pred}).to_csv('Data/Result/predictions_RF.csv',
index=False)
pd.DataFrame({'Actual': y_test, 'SVM_Pred': svm_pred}).to_csv('Data/Result/predictions_SVM.csv',
index=False)
pd.DataFrame({'Actual': y_test, 'DT_Pred': dt_pred}).to_csv('Data/Result/predictions_DT.csv',
index=False)
pd.DataFrame({'Actual': y_test, 'NB_Pred': nb_pred}).to_csv('Data/Result/predictions_NB.csv',
index=False)
pd.DataFrame({'Actual': y_test, 'KNN_Pred': knn_pred}).to_csv('Data/Result/predictions_KNN.csv',
index=False)
pd.DataFrame({'Actual': y_test, 'LinReg_Pred':
linreg_pred}).to_csv('Data/Result/predictions_LinReg.csv', index=False)
pd.DataFrame({'Actual': y_test, 'ANN_Pred': ann_pred}).to_csv('Data/Result/predictions_ANN.csv',
index=False)
```

Step 13: Visualize Key Insights

```
# Gender distribution
sns.countplot(data=nobel, x='sex')
plt.title('Gender Distribution of Nobel Laureates')
plt.show()

# Top 10 countries
nobel['birth_country'].value_counts().head(10).plot(kind='barh')
plt.title('Top 10 Countries with Most Nobel Laureates')
plt.show()

nobel['decade'] = (nobel['year'] // 10) * 10
nobel['usa_born_winners'] = nobel['birth_country'] == 'United States of America'

# USA-born winners by decade
sns.lineplot(data=nobel.groupby('decade')['usa_born_winners'].mean().reset_index(), x='decade',
y='usa_born_winners')
plt.title('Proportion of USA-born Winners per Decade')
plt.show()
```


Gender Distribution of Nobel Laureates
800
600
400
200
0
count
Male
Female
sex


Top 10 Countries with Most Nobel Laureates
Italy
Canada
Netherlands
Russia
Japan
Sweden
France
Germany
United Kingdom
United States of America
birth_country
0
50
100
150
200
250
300



<u>8.The answer of question:</u>

1. What is the name of your data?

Nobel Prize Winners Dataset

2. The source of the data (which database)?

Our World in Data kaggle

3. Link to the original data?

Link: https://www.kaggle.com/datasets/imdevskp/nobel-prize

4. Explain the data in words

This dataset contains information about Nobel Prize winners from 1901 to 2016, including their names, gender, birth countries, award years, categories, and affiliated organizations.

5. Is it a regression or classification problem?

It is a **binary classification problem**. We are predicting whether the winner is female (1) or male (0).

6. How many attributes?

We used 4 attributes for modeling:

- year
- category_encoded
- country_encoded
- sex_encoded

7. How many samples?

There are **911 samples** in total.

8. What are the properties of the data (statistics)?

Descriptive statistics for numerical columns:

- **year**: mean = 1975.5, std = 32.7
- **category_encoded**: 6 unique values
- **sex_encoded**: 2 values (0 for male, 1 for female)
- **country_encoded**: multiple countries encoded as integers

9. Are there any missing data? How did you fill in the missing values?

Yes, there were missing values in sex and birth_country.
We filled them using the **most frequent value (mode)**.

10. Visualize the data

We used seaborn and matplotlib to create:

- Count plots for categories by gender
- Line plot of awards per decade

11. Did you normalize or standardize any of your data? Why?

Yes, we **standardized** the features using StandardScaler because the scale of features like year is significantly different from encoded categorical values.

12. What type of preprocessing did you apply to your data? List everything and explain why.

- Filled missing values (to avoid errors during training)
- Dropped irrelevant columns (to reduce noise)
- Created new features (female_winner, decade)
- Label encoding for categorical features
- Standardization using StandardScaler
  Each step improves model accuracy and performance.

13. How did you divide the train and test data? What are the proportions?

Used train_test_split with an 80/20 split.

- 80% for training
- 20% for testing

14. Apply all the machine learning models you have learned in this course to your data and report the results. What is the best/worst performing model? Why?

Models applied:

- Logistic Regression
- Random Forest
- K-Nearest Neighbors
- Support Vector Machine

- ANN
- Linear regression
- Decision tree

Best Model: Random Forest — because it handles nonlinear relationships well.
Worst Model: KNN — possibly due to feature scale sensitivity and data imbalance.

15. The accuracy of all models using tables and figures

Model	Accuracy
Super Vector Machine	89%
Decision Tree	85%
Random Forest	90%
K-Nearest Neighbors	83%
Naive Bayes	81%
Linear Regression	79%
ANN (MLP Classifier)	88%

16. Advanced visualization using seaborn and techniques (bonus)

Answer:

- Used seaborn.heatmap for correlations
- countplot, pairplot, barplot, lineplot
- Confusion matrix visualizations

17. What is the reason you picked up this data? What is the importance of your data in reality, and what is the importance of your best-performing model? Is there any insight you could share from the data and the model?

I picked this data because it connects history, global achievement, and diversity. The Nobel Prize is one of the most prestigious honors, and exploring patterns of gender and country representation reveals insightful trends. The data reflects how global society values scientific and cultural contributions, and highlights underrepresentation, particularly among female laureates. Our best-performing model (Random Forest) helps us predict whether a winner is female based on year, category, and birth country. This insight could support further research on equity and policy development. The analysis shows a gradual increase in female winners, especially in Peace and Literature. The data also shows dominance of specific countries, raising questions on global access to opportunities. By modeling and visualizing these insights, we contribute to understanding long-term trends and supporting inclusive recognition of talent worldwide.

.