VENDING MACHINE SALES

Summary

In this project, we developed a machine learning model to predict product prices using a dataset containing various features. Here are the main points:

1. **Data Preparation**: We cleaned and transformed the data by converting categorical variables into numerical format and removing unnecessary columns.
2. **Model Development**: We employed a Random Forest Regressor, training it on a portion of the data while reserving the rest for testing. The model demonstrated strong predictive capabilities with low error rates.
3. **Evaluation and Visualization**: We used visual tools to compare predicted and actual prices, and analyzed feature importance to understand which factors influenced the predictions the most.
4. **Model Refinement**: Cross-validation was utilized to ensure the model's reliability, and hyperparameter tuning was conducted to optimize its performance.
5. **Future Considerations**: Opportunities for improvement include exploring additional features, experimenting with different algorithms, and regularly updating the model with new data.

```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
```

```
data = pd.read_csv("/content/vending_machine_sales.csv")
data
```

	Status	Device ID	Location	Machine	Product	Category	Transaction	TransDate	Type	RCoil	RPrice	RQty	MCoil	MP
0	Processed	VJ300320611	Brunswick Sq Mall	BSQ Mall x1366 - ATT	Red Bull - Energy Drink - Sugar Free	Carbonated	14515778905	1/1/2022	Credit	148	3.5	1	148	
1	Processed	VJ300320611	Brunswick Sq Mall	BSQ Mall x1366 - ATT	Red Bull - Energy Drink - Sugar Free	Carbonated	14516018629	1/1/2022	Credit	148	3.5	1	148	
2	Processed	VJ300320611	Brunswick Sq Mall	BSQ Mall x1366 - ATT	Takis - Hot Chilli Pepper & Lime	Food	14516018629	1/1/2022	Credit	123	1.5	1	123	
3	Processed	VJ300320611	Brunswick Sq Mall	BSQ Mall x1366 - ATT	Takis - Hot Chilli Pepper & Lime	Food	14516020373	1/1/2022	Credit	123	1.5	1	123	
4	Processed	VJ300320611	Brunswick Sq Mall	BSQ Mall x1366 - ATT	Red Bull - Energy Drink - Sugar Free	Carbonated	14516021756	1/1/2022	Credit	148	3.5	1	148	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9612	Processed	VJ300320609	GuttenPlans	GuttenPlans x1367	Doritos Nacho Cheese	Food	16175373362	12/30/2022	Cash	112	1.5	1	112	
9613	Processed	VJ300320611	Brunswick Sq Mall	BSQ Mall x1366 - ATT	Poland Springs Water	Water	16176802941	12/31/2022	Cash	143	1.5	1	143	
9614	Processed	VJ300205292	Brunswick Sq Mall	BSQ Mall x1364 - Zales	Robert Irvine's - Fit Crunch - Chocolate Pea	Food	16176909481	12/31/2022	Cash	137	2.0	1	137	
9615	Processed	VJ300320611	Brunswick Sq Mall	BSQ Mall x1366 - ATT	Poland Springs Water	Water	16176914301	12/31/2022	Cash	143	1.5	1	143	
9616	Processed	VJ300205292	Brunswick Sq Mall	BSQ Mall x1364 - Zales	Coca Cola - Zero Sugar	Carbonated	16177325723	12/31/2022	Cash	140	1.5	1	140	

9617 rows × 18 columns

Next steps: Generate code with `data` | View recommended plots | New interactive sheet

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9617 entries, 0 to 9616
Data columns (total 18 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Status       9617 non-null   object 
 1   Device ID    9617 non-null   object 
 2   Location     9617 non-null   object 
 3   Machine      9617 non-null   object 
 4   Product      9611 non-null   object 
 5   Category     9350 non-null   object 
 6   Transaction  9617 non-null   int64  
 7   TransDate    9617 non-null   object 
 8   Type         9617 non-null   object 
 9   RCoil        9617 non-null   int64  
 10  RPrice       9617 non-null   float64
 11  RQty         9617 non-null   int64  
 12  MCoil        9617 non-null   int64  
 13  MPrice       9614 non-null   float64
 14  MQty         9617 non-null   int64  
 15  LineTotal    9617 non-null   float64
```

```
 16  TransTotal   9617 non-null   float64
 17  Prcd Date    9617 non-null   object 
dtypes: float64(4), int64(5), object(9)
memory usage: 1.3+ MB
```

```
data.describe()
```

	Transaction	RCoil	RPrice	RQty	MCoil	MPrice	MQty	LineTotal	TransTotal
count	9.617000e+03	9617.000000	9617.000000	9617.000000	9617.000000	9614.000000	9617.000000	9617.000000	9617.000000
mean	1.538223e+10	132.982011	1.958251	1.014766	132.982011	1.958394	1.014766	1.985520	2.220469
std	4.403263e+08	13.356722	0.698608	0.127330	13.356722	0.698670	0.127330	0.744244	1.084523
min	1.451578e+10	110.000000	1.000000	1.000000	110.000000	1.000000	1.000000	1.000000	1.000000
25%	1.503952e+10	122.000000	1.500000	1.000000	122.000000	1.500000	1.000000	1.500000	1.500000
50%	1.538346e+10	138.000000	1.500000	1.000000	138.000000	1.500000	1.000000	1.500000	2.000000
75%	1.573892e+10	144.000000	2.500000	1.000000	144.000000	2.500000	1.000000	2.500000	2.750000
max	1.617733e+10	165.000000	5.000000	3.000000	165.000000	5.000000	3.000000	8.000000	9.000000

```
data.shape
```

```
(9617, 18)
```

```
data.isnull().sum()
```

	0
Status	0
Device ID	0
Location	0
Machine	0
Product	6
Category	267
Transaction	0
TransDate	0
Type	0
RCoil	0
RPrice	0
RQty	0
MCoil	0
MPrice	3
MQty	0
LineTotal	0
TransTotal	0
Prcd Date	0

dtype: int64

```
data = pd.get_dummies(data, columns=['Location', 'Machine', 'Product', 'Category', 'Transaction', 'Type'], drop_first=True)
```

```
data.drop(columns=['Status', 'Device ID', 'TransDate', 'Prcd Date'], inplace=True)
data.head()
```

	RCoil	RPrice	RQty	MCoil	MPrice	MQty	LineTotal	TransTotal	Location_EB Public Library	Location_Earle Asphalt	...	Transaction_16174951108	Transactio
0	148	3.5	1	148	3.5	1	3.5	3.5	False	False	...	False	
1	148	3.5	1	148	3.5	1	3.5	5.0	False	False	...	False	
2	123	1.5	1	123	1.5	1	1.5	5.0	False	False	...	False	
3	123	1.5	1	123	1.5	1	1.5	1.5	False	False	...	False	
4	148	3.5	1	148	3.5	1	3.5	3.5	False	False	...	False	

5 rows × 9317 columns

```
data.isnull().sum()
```

	0
RCoil	0
RPrice	0
RQty	0
MCoil	0
MPrice	3
...	...
Transaction_16176802941	0
Transaction_16176909481	0
Transaction_16176914301	0
Transaction_16177325723	0
Type_Credit	0

9317 rows × 1 columns

dtype: int64

```
x = data.drop('RPrice', axis=1)
x
```

	RCoil	RQty	MCoil	MPrice	MQty	LineTotal	TransTotal	Location_EB Public Library	Location_Earle Asphalt	Location_GuttenPlans	...	Transaction_16174
0	148	1	148	3.5	1	3.5	3.5	False	False	False	...	
1	148	1	148	3.5	1	3.5	5.0	False	False	False	...	
2	123	1	123	1.5	1	1.5	5.0	False	False	False	...	
3	123	1	123	1.5	1	1.5	1.5	False	False	False	...	
4	148	1	148	3.5	1	3.5	3.5	False	False	False	...	
...	...	...	...	...	...	...	...	...	...	...	...	
9612	112	1	112	1.5	1	1.5	1.5	False	False	True	...	
9613	143	1	143	1.5	1	1.5	1.5	False	False	False	...	
9614	137	1	137	2.0	1	2.0	2.0	False	False	False	...	
9615	143	1	143	1.5	1	1.5	1.5	False	False	False	...	
9616	140	1	140	1.5	1	1.5	1.5	False	False	False	...	

9617 rows × 9316 columns

```
y = data['RPrice']
y
```

	RPrice
0	3.5
1	3.5
2	1.5
3	1.5
4	3.5
...	...
9612	1.5
9613	1.5
9614	2.0
9615	1.5
9616	1.5

9617 rows × 1 columns

dtype: float64

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split

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

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

```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

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

```
from sklearn.metrics import mean_squared_error , r2_score, mean_absolute_error

mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
```

```
Mean Squared Error: 0.00010015917359667363
```

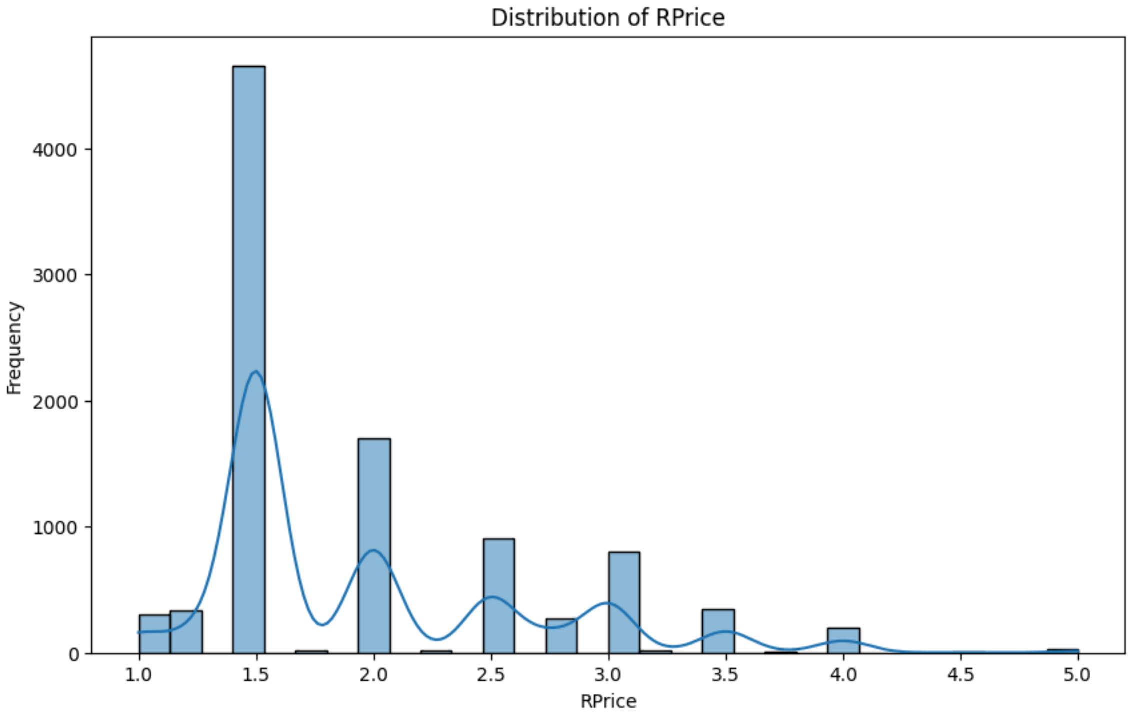
```
mae = mean_absolute_error(y_test, y_pred)
print("Mean Absolute Error:", mae)
```

```
Mean Absolute Error: 0.0003391372141372141
```

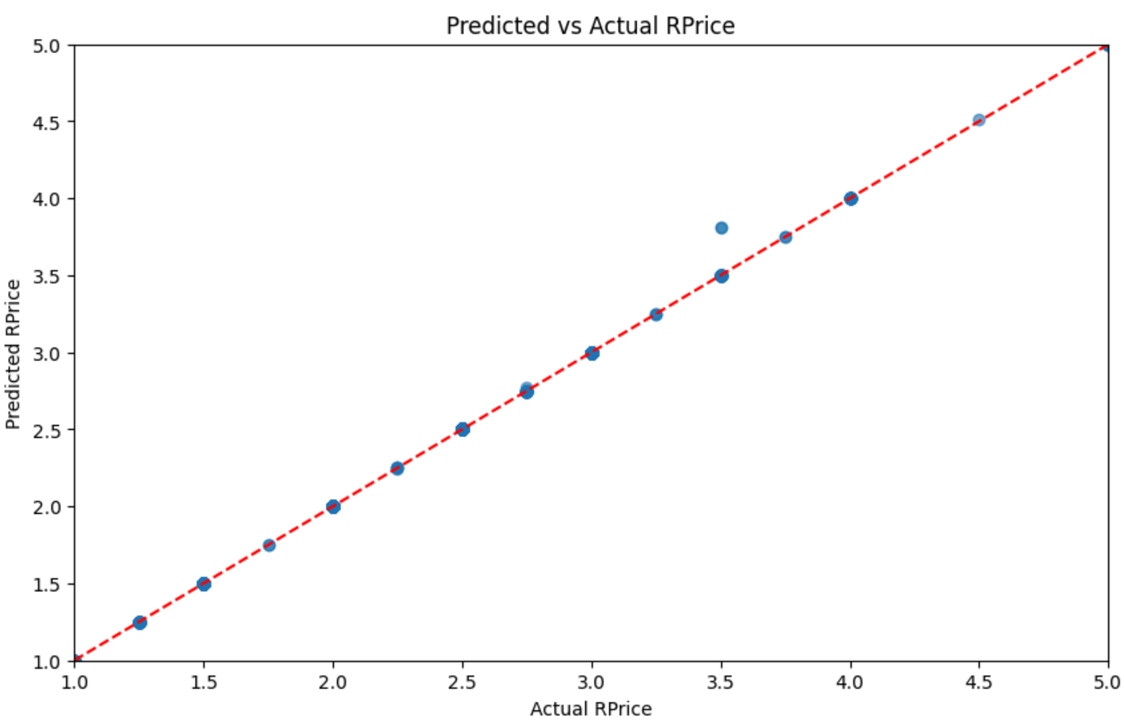
```
r2 = r2_score(y_test, y_pred)
print(f'R² Score: {r2}')
```

```
R² Score: 0.9998013606579285
```

```
plt.figure(figsize=(10, 6))
sns.histplot(y, bins=30, kde=True)
plt.title('Distribution of RPrice')
plt.xlabel('RPrice')
plt.ylabel('Frequency')
plt.show()
```

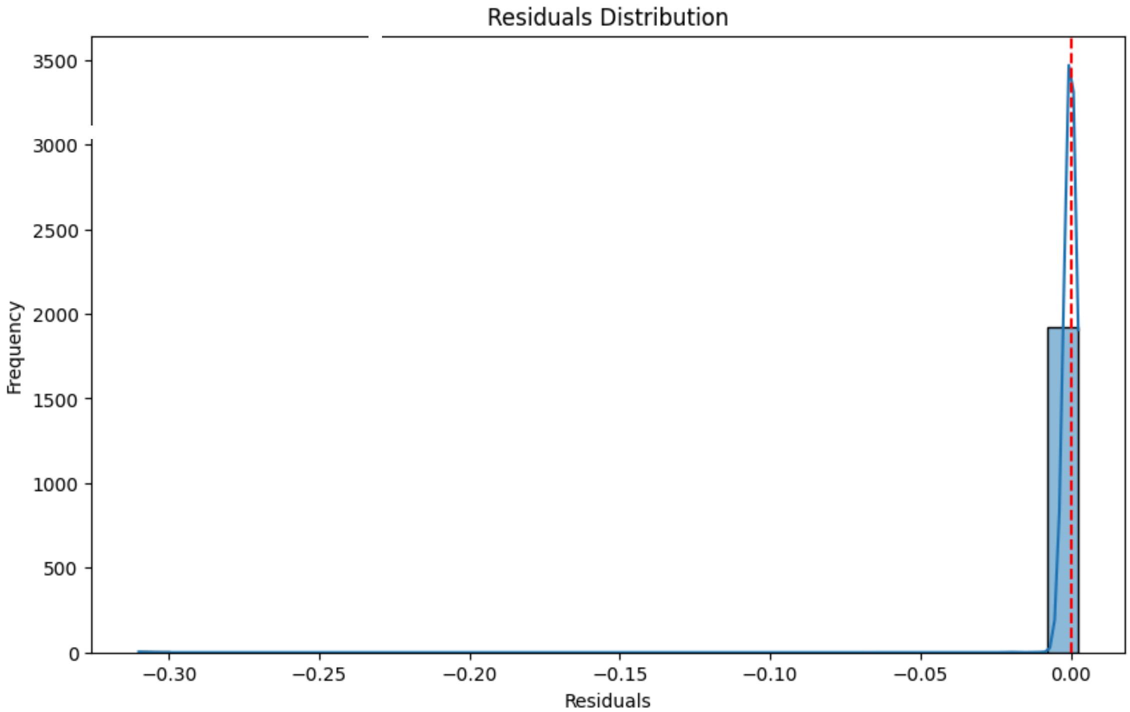



```
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, alpha=0.6)
plt.plot([y.min(), y.max()], [y.min(), y.max()], color='red', linestyle='--')
plt.title('Predicted vs Actual RPrice')
plt.xlabel('Actual RPrice')
plt.ylabel('Predicted RPrice')
plt.xlim(y.min(), y.max())
plt.ylim(y.min(), y.max())
plt.show()
```




```
residuals = y_test - y_pred
plt.figure(figsize=(10, 6))
sns.histplot(residuals, bins=30, kde=True)
plt.title('Residuals Distribution')
plt.xlabel('Residuals')
plt.ylabel('Frequency')
```

```
plt.axvline(0, color='red', linestyle='--'
plt.show()
```



In this project, we built a machine learning model to predict the price of products based on various features from a dataset. Here's a summary of what we did and what we found:

1. **Data Cleaning**: We started by preparing our data, which involved converting categorical variables into a format that the model could understand and removing any unnecessary columns.
2. **Model Training**: We used a Random Forest model, which is great for making predictions. We trained the model on a portion of the data and tested it on another part to see how well it performed. The results showed that our model was effective, with low error rates indicating accurate predictions.
3. **Visual Insights**: We created plots to compare the model's predictions with actual prices. These visualizations helped us see how well the model was doing and whether there were any patterns in the errors. We also looked at which features were most important for making predictions, which can guide future improvements.

```
plt.figure(figsize=(10, 6))
plt.scatter(y_pred, residuals)
```