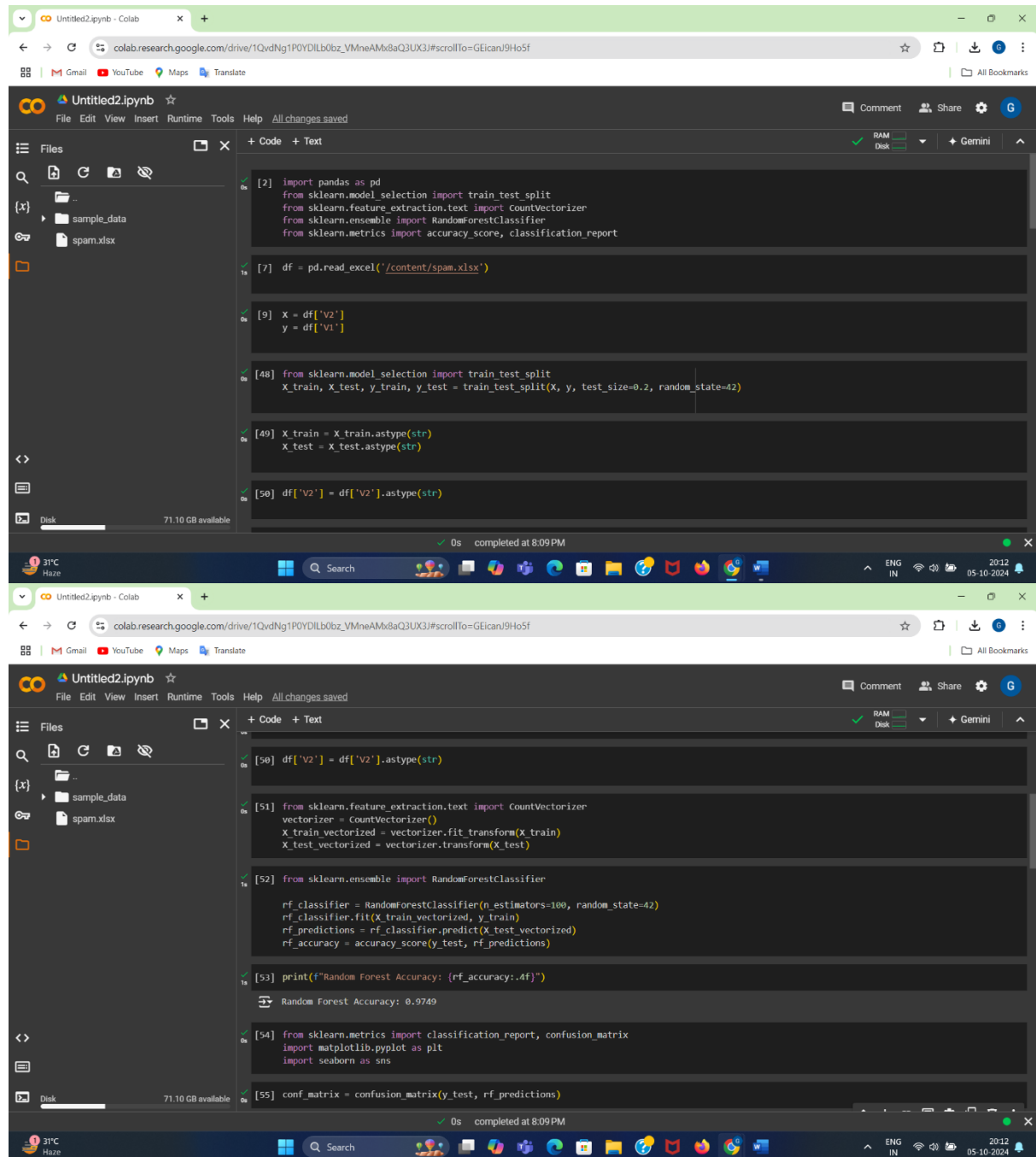


Spam Detection Project

I have used the random forest classification model to detect spam vs ham messages.



The image displays two screenshots of a Google Colab notebook titled 'Untitled2.ipynb'. The notebook is open in a web browser, showing the code editor and file explorer. The file explorer on the left shows a folder named 'sample_data' containing a file named 'spam.xlsx'. The code editor on the right shows the following code:

```
[2] import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

[7] df = pd.read_excel('/content/spam.xlsx')

[9] X = df['v2']
y = df['v1']

[48] 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)

[49] X_train = X_train.astype(str)
X_test = X_test.astype(str)

[50] df['v2'] = df['v2'].astype(str)

[50] df['v2'] = df['v2'].astype(str)

[51] from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)

[52] from sklearn.ensemble import RandomForestClassifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
rf_classifier.fit(X_train_vectorized, y_train)
rf_predictions = rf_classifier.predict(X_test_vectorized)
rf_accuracy = accuracy_score(y_test, rf_predictions)

[53] print(f"Random Forest Accuracy: {rf_accuracy:.4f}")
Random Forest Accuracy: 0.9749

[54] from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

[55] conf_matrix = confusion_matrix(y_test, rf_predictions)
```

The notebook interface shows the code is executed successfully, with a status bar indicating '0s completed at 8:09 PM'. The bottom of the notebook shows a Windows taskbar with the date '05-10-2024' and time '20:12'.

The top screenshot shows the following code and output:

```
[54] from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

[55] conf_matrix = confusion_matrix(y_test, rf_predictions)

[56] plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['Ham', 'Spam'], yticklabels=['Ham', 'Spam'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix for Random Forest')
plt.show()

[57] class_report = classification_report(y_test, rf_predictions)
print("Classification Report:\n", class_report)
```

		precision	recall	f1-score	support
ham	0.97	1.00	0.99	0.99	965
spam	1.00	0.81	0.90	0.85	150
accuracy			0.97		1115
macro avg	0.99	0.91	0.94	0.94	1115
weighted avg	0.98	0.97	0.97	0.97	1115

The bottom screenshot shows the following code and output:

```
feature_importances = rf_classifier.feature_importances_
features = vectorizer.get_feature_names_out()
feature_importance_df = pd.DataFrame({'feature': features, 'importance': feature_importances})
feature_importance_df = feature_importance_df.sort_values(by='importance', ascending=False).head(10)

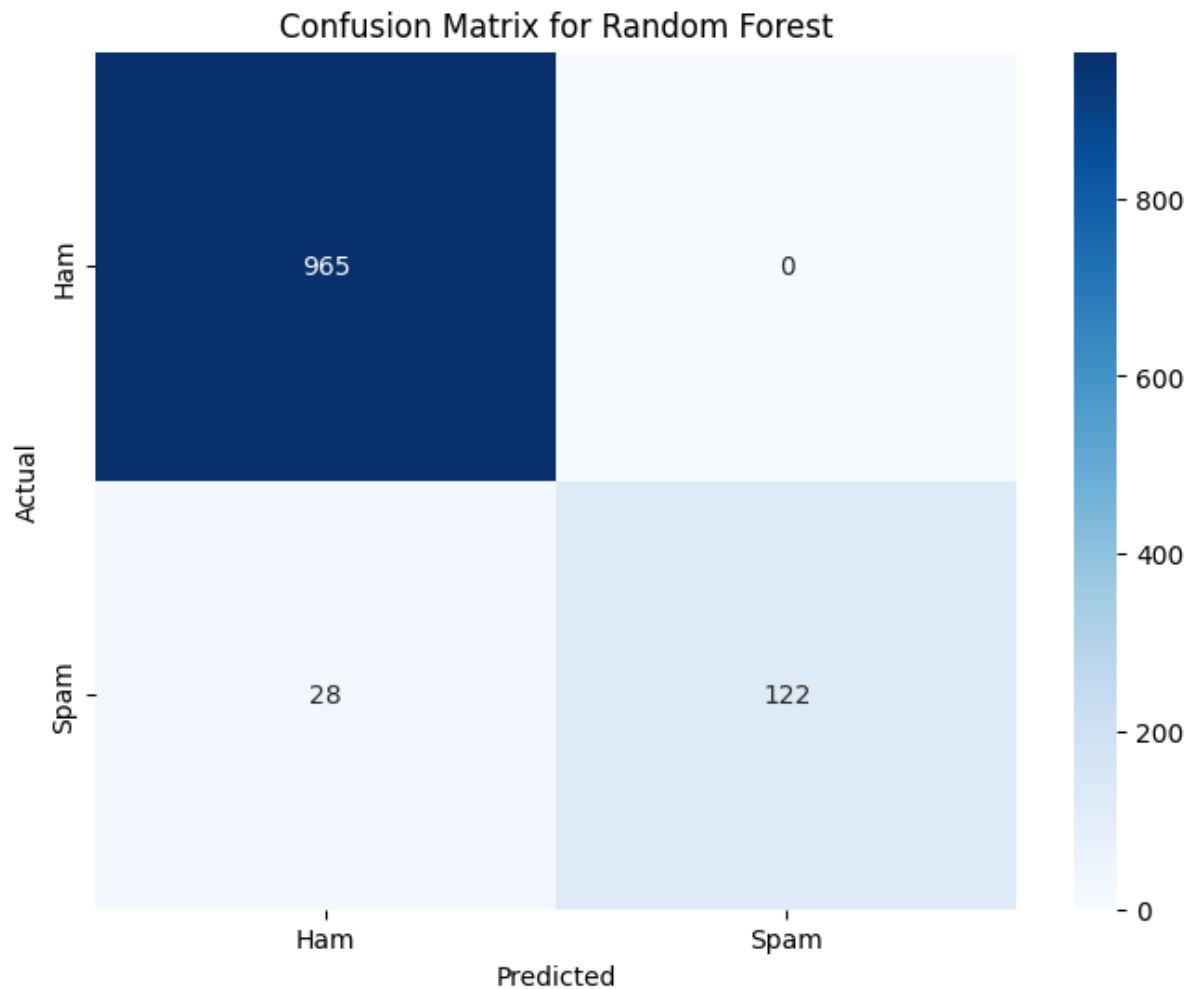
print("Top 10 Important Features:\n", feature_importance_df)
```

feature	importance
call	0.029093
txt	0.026982
claim	0.022545
free	0.022163
www	0.019241
150p	0.017343
mobile	0.015245
uk	0.013451
stop	0.010761
prize	0.010514

The random forest model achieved an accuracy of 97.49%.

Conclusions:

1) This confusion matrix was generated:



The confusion matrix shows excellent performance in classifying ham messages (all 965/965 correct) and decent performance for spam (122/150).

2) Classification report.

	precision	recall	F1-score	Support
ham	0.97	1.00	0.99	965
spam	1.00	0.81	0.90	150
accuracy			0.97	1115
macro avg	0.99	0.91	0.94	1115
weight avg	0.98	0.97	0.97	1115

- The model achieves 97% precision and 100% recall for ham messages.
- For spam messages, it has 100% precision but 81% recall.
- Overall accuracy is 98%, so very strong performance.

3) Top 10 important features.

	features	importance
1609	call	0.029093
7066	txt	0.026982
1828	claim	0.022545
3001	free	0.022163
7612	www	0.019241
305	150p	0.017343
4526	mobile	0.015245

7091	uk	0.013451
6468	stop	0.010761
5403	prize	0.010514

The top features for classification include words commonly associated with spam, such as "txt," "call," "claim," and "prize."