Forest Cover Type Prediction - Internship Project

Data Initialization

Dependencies

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, classification_report, f1_score
```

In [2]:
```
pd.set_option('display.max_columns', None)
```

Merging two data

In [3]:
```
data1 = pd.read_csv('train.csv',index_col="Id") # Got from internship
data2 = pd.read_csv('covtype.csv') # got from internet
forestData = pd.concat([data2,data1],ignore_index=True)
```

Viewing Data

In [4]:
```
forestData.head()
```

Out[4]:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizo
0	2596	51	3	258	0	
1	2590	56	2	212	-6	
2	2804	139	9	268	65	
3	2785	155	18	242	118	
4	2595	45	2	153	-1	

Shape

In [5]:
```
print(f"No. of rows: {forestData.shape[0]}")
print(f"No. of cols: {forestData.shape[1]}")
```

```
No. of rows: 596132
No. of cols: 55
```

Data Info

In [6]:
```
forestData.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 596132 entries, 0 to 596131
Data columns (total 55 columns):
 #   Column                              Non-Null Count   Dtype
---  ------                              --------------   -----
 0   Elevation                           596132 non-null  int64
 1   Aspect                              596132 non-null  int64
 2   Slope                               596132 non-null  int64
 3   Horizontal_Distance_To_Hydrology    596132 non-null  int64
 4   Vertical_Distance_To_Hydrology      596132 non-null  int64
 5   Horizontal_Distance_To_Roadways     596132 non-null  int64
 6   Hillshade_9am                       596132 non-null  int64
 7   Hillshade_Noon                      596132 non-null  int64
 8   Hillshade_3pm                       596132 non-null  int64
 9   Horizontal_Distance_To_Fire_Points  596132 non-null  int64
 10  Wilderness_Area1                    596132 non-null  int64
 11  Wilderness_Area2                    596132 non-null  int64
 12  Wilderness_Area3                    596132 non-null  int64
 13  Wilderness_Area4                    596132 non-null  int64
 14  Soil_Type1                          596132 non-null  int64
 15  Soil_Type2                          596132 non-null  int64
 16  Soil_Type3                          596132 non-null  int64
 17  Soil_Type4                          596132 non-null  int64
 18  Soil_Type5                          596132 non-null  int64
 19  Soil_Type6                          596132 non-null  int64
 20  Soil_Type7                          596132 non-null  int64
 21  Soil_Type8                          596132 non-null  int64
 22  Soil_Type9                          596132 non-null  int64
 23  Soil_Type10                         596132 non-null  int64
 24  Soil_Type11                         596132 non-null  int64
 25  Soil_Type12                         596132 non-null  int64
 26  Soil_Type13                         596132 non-null  int64
 27  Soil_Type14                         596132 non-null  int64
 28  Soil_Type15                         596132 non-null  int64
 29  Soil_Type16                         596132 non-null  int64
 30  Soil_Type17                         596132 non-null  int64
 31  Soil_Type18                         596132 non-null  int64
 32  Soil_Type19                         596132 non-null  int64
 33  Soil_Type20                         596132 non-null  int64
 34  Soil_Type21                         596132 non-null  int64
 35  Soil_Type22                         596132 non-null  int64
 36  Soil_Type23                         596132 non-null  int64
 37  Soil_Type24                         596132 non-null  int64
 38  Soil_Type25                         596132 non-null  int64
 39  Soil_Type26                         596132 non-null  int64
 40  Soil_Type27                         596132 non-null  int64
 41  Soil_Type28                         596132 non-null  int64
 42  Soil_Type29                         596132 non-null  int64
 43  Soil_Type30                         596132 non-null  int64
 44  Soil_Type31                         596132 non-null  int64
 45  Soil_Type32                         596132 non-null  int64
 46  Soil_Type33                         596132 non-null  int64
 47  Soil_Type34                         596132 non-null  int64
 48  Soil_Type35                         596132 non-null  int64
 49  Soil_Type36                         596132 non-null  int64
 50  Soil_Type37                         596132 non-null  int64
 51  Soil_Type38                         596132 non-null  int64
 52  Soil_Type39                         596132 non-null  int64
 53  Soil_Type40                         596132 non-null  int64
 54  Cover_Type                          596132 non-null  int64
dtypes: int64(55)
memory usage: 250.1 MB
```

In [7]:

```
forestData.describe()
```

Out[7]:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distan
count	596132.000000	596132.000000	596132.000000	596132.000000	
mean	2954.037879	155.682674	14.164522	268.357052	
std	286.213696	111.867752	7.523713	212.590510	
min	1859.000000	0.000000	0.000000	0.000000	
25%	2801.000000	59.000000	9.000000	108.000000	
50%	2993.000000	127.000000	13.000000	218.000000	
75%	3163.000000	260.000000	19.000000	384.000000	
max	3858.000000	360.000000	66.000000	1397.000000	

Checking for any Null values

In [8]:
```
forestData.isna().any()
```

Out[8]:
```
Elevation                             False
Aspect                                False
Slope                                 False
Horizontal_Distance_To_Hydrology      False
Vertical_Distance_To_Hydrology        False
Horizontal_Distance_To_Roadways       False
Hillshade_9am                         False
Hillshade_Noon                        False
Hillshade_3pm                         False
Horizontal_Distance_To_Fire_Points    False
Wilderness_Area1                      False
Wilderness_Area2                      False
Wilderness_Area3                      False
Wilderness_Area4                      False
Soil_Type1                            False
Soil_Type2                            False
Soil_Type3                            False
Soil_Type4                            False
Soil_Type5                            False
Soil_Type6                            False
Soil_Type7                            False
Soil_Type8                            False
Soil_Type9                            False
Soil_Type10                           False
Soil_Type11                           False
Soil_Type12                           False
Soil_Type13                           False
Soil_Type14                           False
Soil_Type15                           False
Soil_Type16                           False
Soil_Type17                           False
Soil_Type18                           False
Soil_Type19                           False
Soil_Type20                           False
Soil_Type21                           False
Soil_Type22                           False
Soil_Type23                           False
Soil_Type24                           False
Soil_Type25                           False
Soil_Type26                           False
Soil_Type27                           False
Soil_Type28                           False
Soil_Type29                           False
Soil_Type30                           False
Soil_Type31                           False
Soil_Type32                           False
Soil_Type33                           False
Soil_Type34                           False
Soil_Type35                           False
Soil_Type36                           False
Soil_Type37                           False
Soil_Type38                           False
Soil_Type39                           False
Soil_Type40                           False
Cover_Type                            False
dtype: bool
```

Columns in the data

In [9]:
```
column = forestData.columns
column
```

Out[9]:
```
Index(['Elevation', 'Aspect', 'Slope', 'Horizontal_Distance_To_Hydrology',
       'Vertical_Distance_To_Hydrology', 'Horizontal_Distance_To_Roadways',
       'Hillshade_9am', 'Hillshade_Noon', 'Hillshade_3pm',
       'Horizontal_Distance_To_Fire_Points', 'Wilderness_Area1',
       'Wilderness_Area2', 'Wilderness_Area3', 'Wilderness_Area4',
       'Soil_Type1', 'Soil_Type2', 'Soil_Type3', 'Soil_Type4', 'Soil_Type5',
       'Soil_Type6', 'Soil_Type7', 'Soil_Type8', 'Soil_Type9', 'Soil_Type10',
       'Soil_Type11', 'Soil_Type12', 'Soil_Type13', 'Soil_Type14',
       'Soil_Type15', 'Soil_Type16', 'Soil_Type17', 'Soil_Type18',
       'Soil_Type19', 'Soil_Type20', 'Soil_Type21', 'Soil_Type22',
       'Soil_Type23', 'Soil_Type24', 'Soil_Type25', 'Soil_Type26',
       'Soil_Type27', 'Soil_Type28', 'Soil_Type29', 'Soil_Type30',
       'Soil_Type31', 'Soil_Type32', 'Soil_Type33', 'Soil_Type34',
       'Soil_Type35', 'Soil_Type36', 'Soil_Type37', 'Soil_Type38',
       'Soil_Type39', 'Soil_Type40', 'Cover_Type'],
      dtype='object')
```

Note: There are no null values hence theres no need to do data cleaning

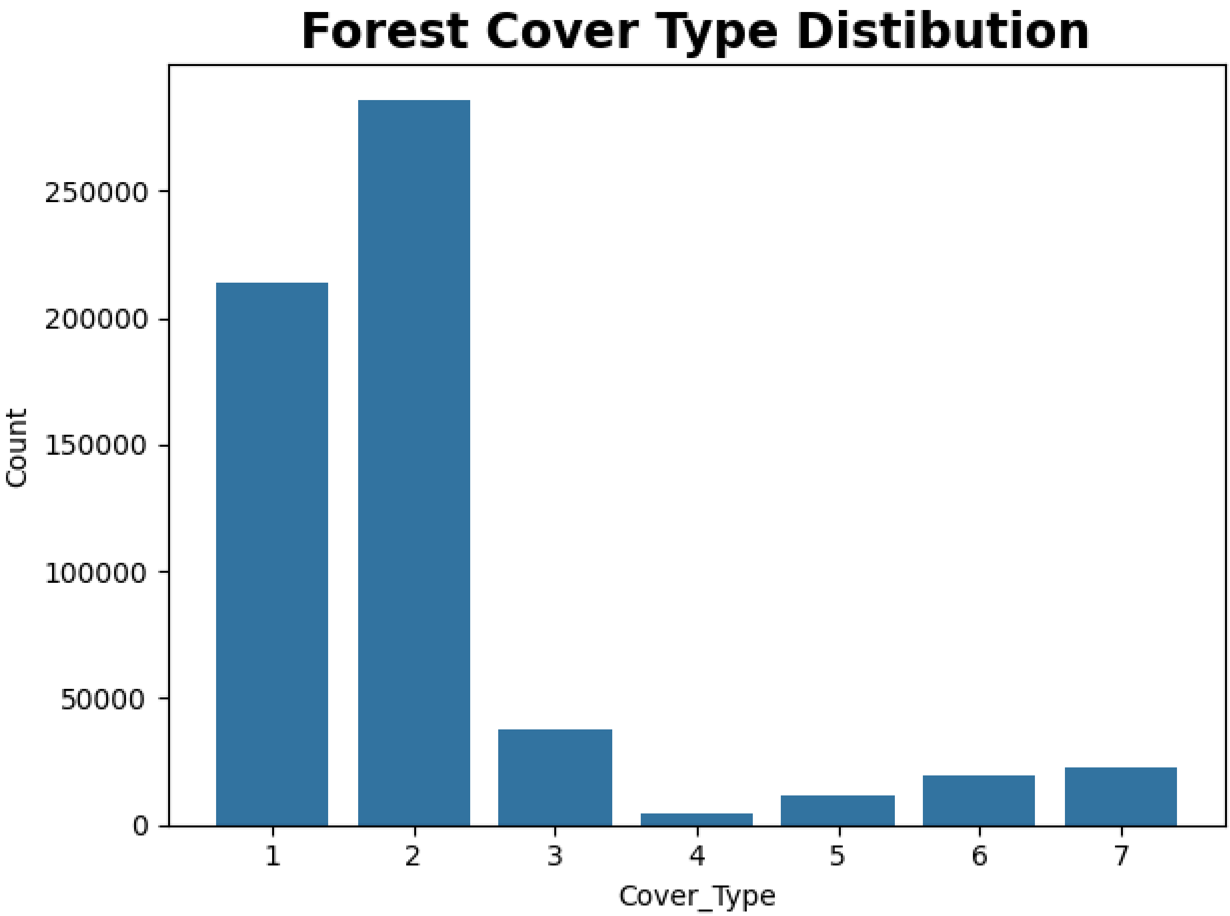
EDA

Target Variable Analysis

- Plot histogram of the forest cover type distribution.
- Check for class imbalance.

In [10]:
```
ax = sns.countplot(data=forestData,x='Cover_Type')
ax.set_xlabel('Cover_Type')
ax.set_ylabel('Count')
ax.set_title('Forest Cover Type Distibution',fontdict={'weight':'600','size':'17'})
plt.tight_layout()
plt.plot()
```

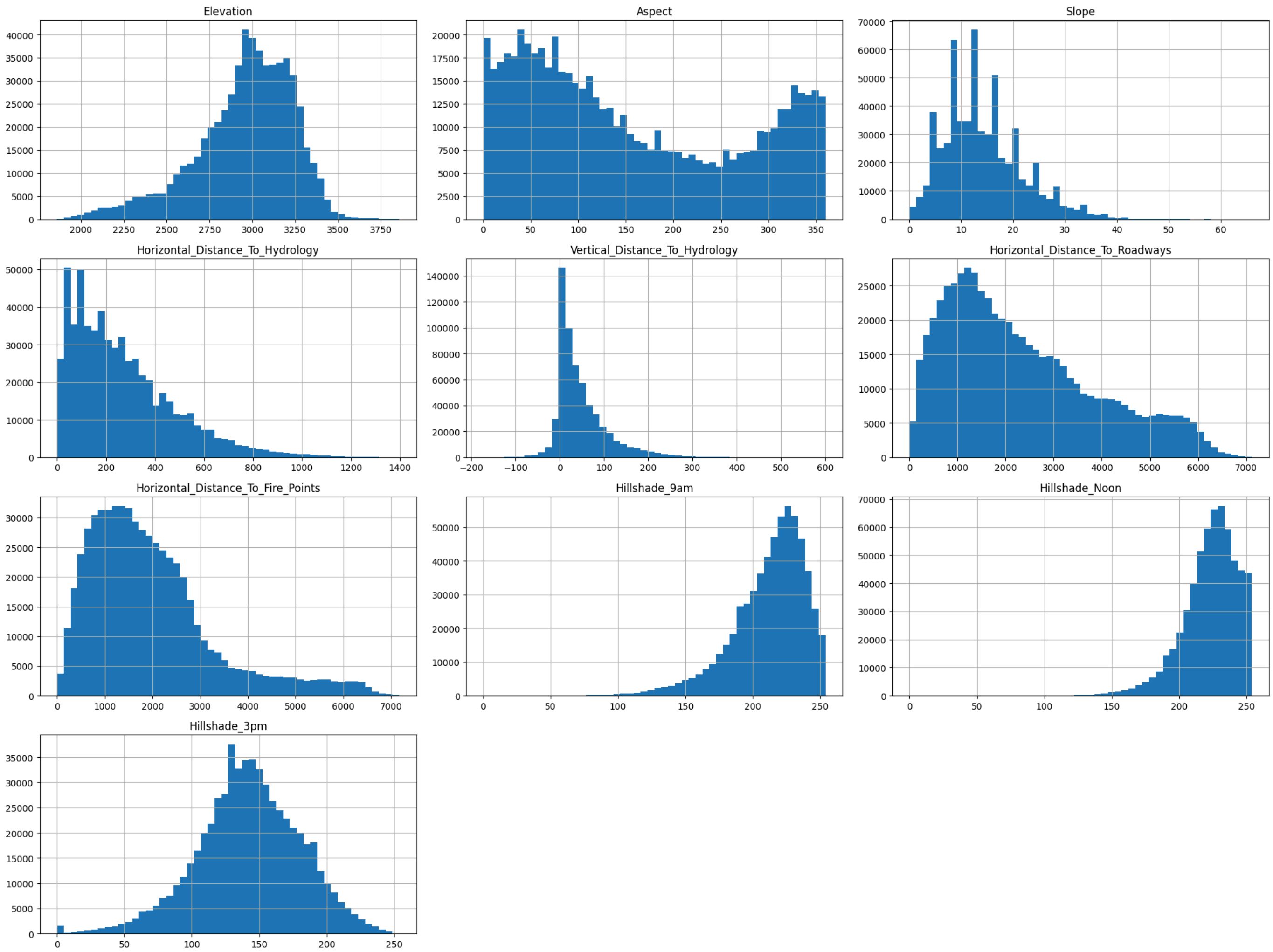
Out[10]:
```
[]
```



Feature Distributions

Plot histograms for each necessary numerical feature

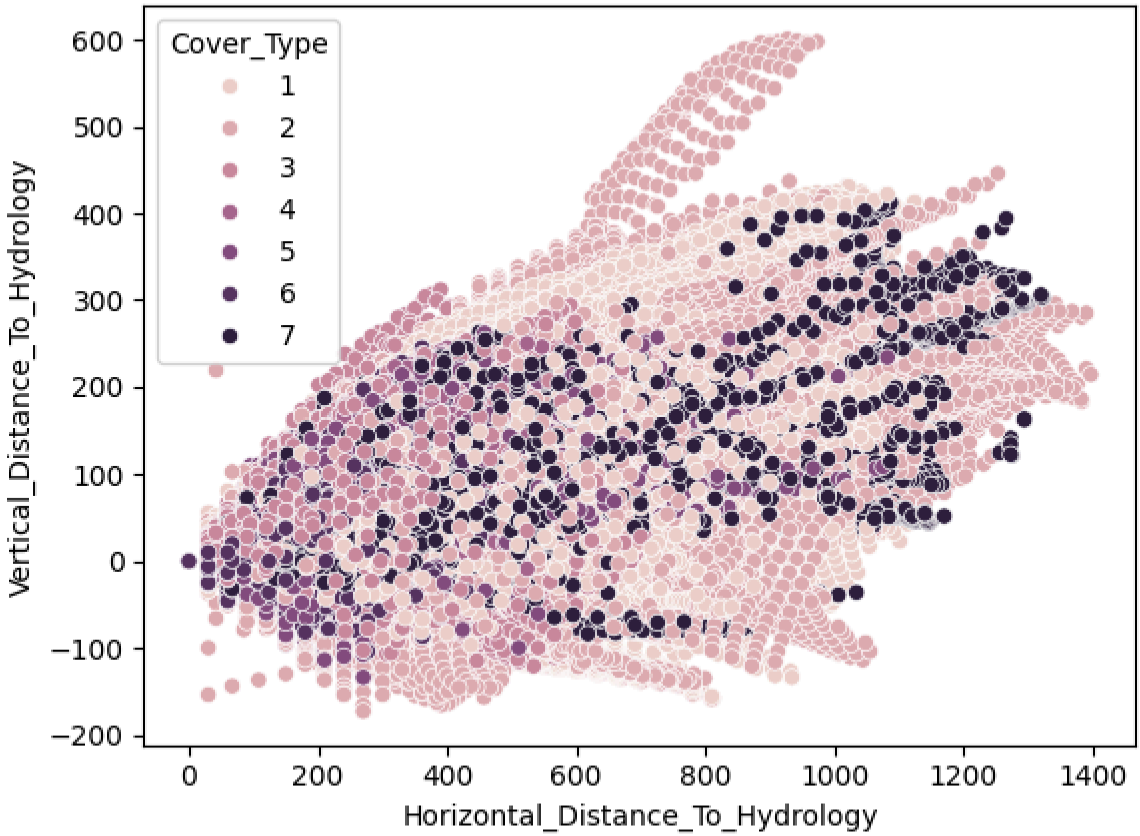
In [11]:
```
forestData[['Elevation','Aspect','Slope','Horizontal_Distance_To_Hydrology','Vertical_Distanc
            'Horizontal_Distance_To_Roadways','Horizontal_Distance_To_Fire_Points','Hillshade_
            'Hillshade_3pm']].hist(bins=50, figsize=(20,15))
plt.tight_layout()
plt.show()
```



Geospatial Relationships

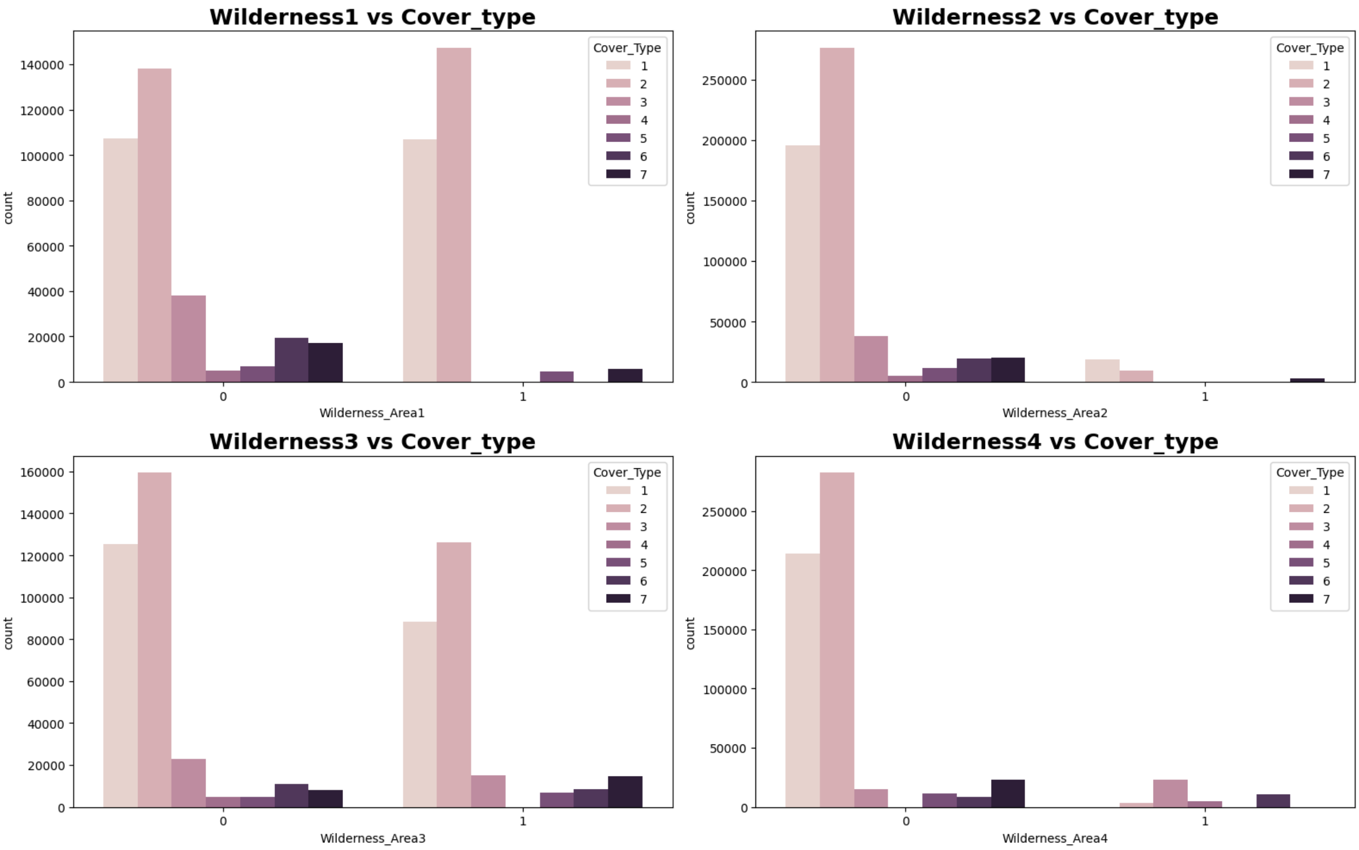
🌍 Since this is geographical data, features like Horizontal_Distance_To_Roadways, Vertical_Distance_To_Hydrology, etc., may relate spatially.

In [12]:
```
sns.scatterplot(x='Horizontal_Distance_To_Hydrology', y='Vertical_Distance_To_Hydrology', hue
plt.show()
```



Wilderness Area vs Cover Type analysis

In [13]:
```
# make it for other wilderness area too
fig,axes = plt.subplots(2,2,figsize = (16,10))
ax1 = sns.countplot(x='Wilderness_Area1', hue='Cover_Type', data=forestData,ax=axes[0,0])
ax2 = sns.countplot(x='Wilderness_Area2', hue='Cover_Type', data=forestData,ax=axes[0,1])
ax3 = sns.countplot(x='Wilderness_Area3', hue='Cover_Type', data=forestData,ax=axes[1,0])
ax4 = sns.countplot(x='Wilderness_Area4', hue='Cover_Type', data=forestData,ax=axes[1,1])
ax1.set_title('Wilderness1 vs Cover_type',fontdict={'size':'18','weight':'600'})
ax2.set_title('Wilderness2 vs Cover_type',fontdict={'size':'18','weight':'600'})
ax3.set_title('Wilderness3 vs Cover_type',fontdict={'size':'18','weight':'600'})
ax4.set_title('Wilderness4 vs Cover_type',fontdict={'size':'18','weight':'600'})
plt.tight_layout()
plt.show()
```



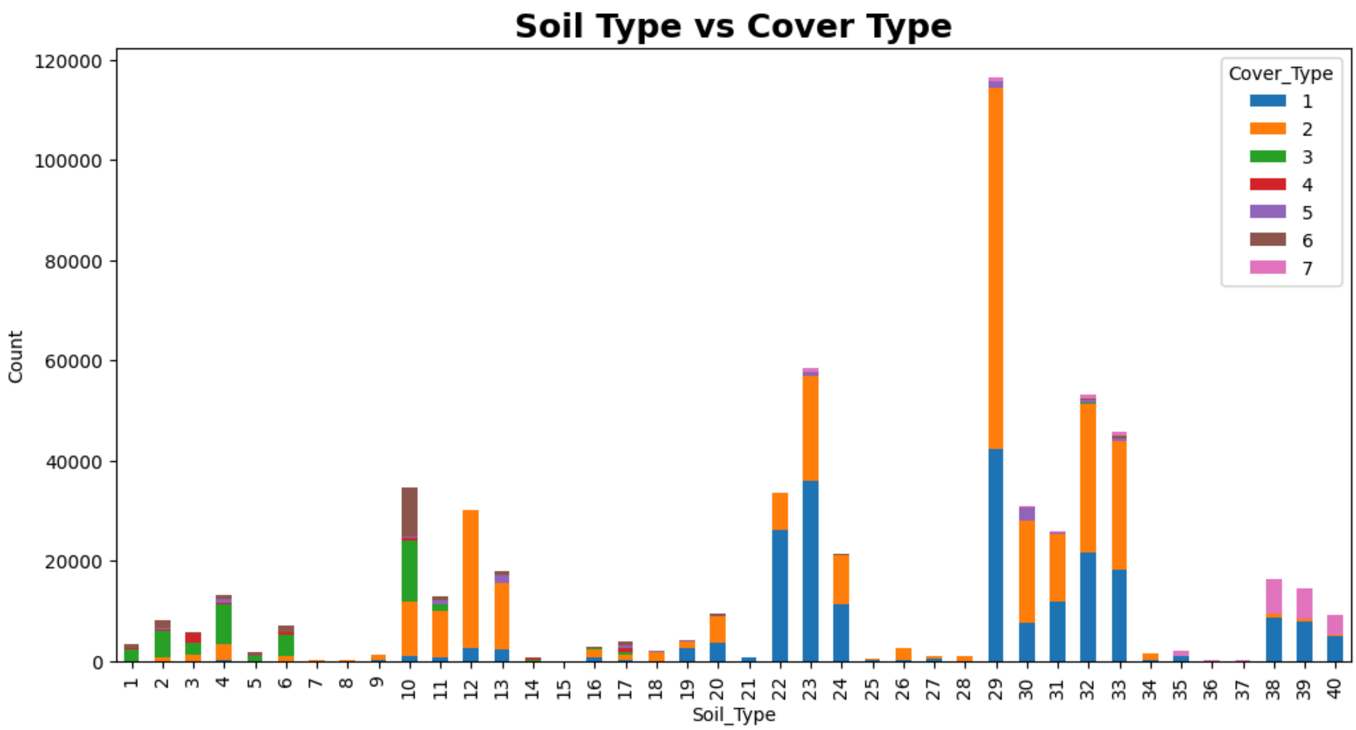
Soil Type vs Cover type relation

Check relationships between soil types and cover types

In [14]:
```python
soil_cols = [f"Soil_Type{i}" for i in range(1,41)]
soil_onehot = forestData[soil_cols]

# get soil type label (e.g. 'Soil_Type7') then convert to integer 7
soil_type_series = soil_onehot.idxmax(axis=1).str.replace('Soil_Type', '').astype(int)

pd.crosstab(soil_type_series, forestData['Cover_Type']).plot(kind='bar', stacked=True, figsiz
plt.xlabel('Soil_Type')
plt.ylabel('Count')
plt.title('Soil Type vs Cover Type',fontdict={'size':'18','weight':'600'})
plt.show()
```



Data Preprocessing

Note: Because we are going to use Tree based models theres no need of Scaling our data

In [15]:
```
X = forestData.drop(['Cover_Type'],axis=1)
y = forestData['Cover_Type']
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=30)
```

Model Selection

Random Forest Implementation

In [16]:
```
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)

rf_classifier.fit(X_train, y_train)
```

Out[16]:
```
▾ RandomForestClassifier
▶ Parameters
```

XGBoost Implementation

In [17]:
```
xgb = XGBClassifier()
xgb_ytrain = y_train.apply(lambda x: x-1)

xgb.fit(X_train,xgb_ytrain)
```

Out[17]:
```
▾ XGBClassifier
▶ Parameters
```

Model Evaluation

In [18]:
```
y_pred = rf_classifier.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)
weighted_f1 = f1_score(y_test, y_pred, average='weighted')

print(f"Accuracy: {accuracy:.2f}")
print(f"Weighted F1 Score: {weighted_f1}")
print("\nClassification Report:\n", classification_rep)
```

```
Accuracy: 0.96
Weighted F1 Score: 0.955195331162959

Classification Report:
               precision    recall  f1-score   support

           1       0.96      0.94      0.95     64145
           2       0.95      0.97      0.96     85642
           3       0.95      0.96      0.96     11388
           4       0.94      0.97      0.96      1487
           5       0.95      0.84      0.89      3417
           6       0.94      0.91      0.93      5938
           7       0.98      0.96      0.97      6823

    accuracy                           0.96    178840
   macro avg       0.95      0.94      0.94    178840
weighted avg       0.96      0.96      0.96    178840
```

In [19]:

```
y_pred = xgb.predict(X_test)
xgb_ytest = y_test.apply(lambda x: x-1)
accuracy = accuracy_score(xgb_ytest, y_pred)
classification_rep = classification_report(xgb_ytest, y_pred)
weighted_f1 = f1_score(xgb_ytest, y_pred, average='weighted')

print(f"Accuracy: {accuracy:.2f}")
print(f"Weighted F1 Score: {weighted_f1}")
print("\nClassification Report:\n", classification_rep)
```

```
Accuracy: 0.87
Weighted F1 Score: 0.8737411073859305

Classification Report:
               precision    recall  f1-score   support

           0       0.87      0.84      0.86     64145
           1       0.87      0.90      0.89     85642
           2       0.90      0.90      0.90     11388
           3       0.91      0.95      0.93      1487
           4       0.89      0.63      0.74      3417
           5       0.85      0.82      0.84      5938
           6       0.94      0.91      0.93      6823

    accuracy                           0.87    178840
   macro avg       0.89      0.85      0.87    178840
weighted avg       0.87      0.87      0.87    178840
```

Note: Random Forest perfomed best in this case