In [11]:

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

In [12]:

```
df = pd.read_excel('loan-train.csv')
df.sample(10)
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Input In [12], in <cell line: 1>()
----> 1 df = pd.read_excel('loan-train.csv')
      2 df.sample(10)

File ~\anaconda3\lib\site-packages\pandas\util\_decorators.py:311, in deprecate_nonke
yword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    305 if len(args) > num_allow_args:
    306     warnings.warn(
    307         msg.format(arguments=arguments),
    308         FutureWarning,
    309         stacklevel=stacklevel,
    310     )
--> 311 return func(*args, **kwargs)

File ~\anaconda3\lib\site-packages\pandas\io\excel\_base.py:457, in read_excel(io, sh
eet_name, header, names, index_col, usecols, squeeze, dtype, engine, converters, true
_values, false_values, skiprows, nrows, na_values, keep_default_na, na_filter, verbos
e, parse_dates, date_parser, thousands, decimal, comment, skipfooter, convert_float,
 mangle_dupe_cols, storage_options)
    455 if not isinstance(io, ExcelFile):
    456     should_close = True
--> 457     io = ExcelFile(io, storage_options=storage_options, engine=engine)
    458 elif engine and engine != io.engine:
    459     raise ValueError(
    460         "Engine should not be specified when passing "
    461         "an ExcelFile - ExcelFile already has the engine set"
    462     )

File ~\anaconda3\lib\site-packages\pandas\io\excel\_base.py:1376, in ExcelFile.__init
__(self, path_or_buffer, engine, storage_options)
   1374     ext = "xls"
   1375 else:
-> 1376     ext = inspect_excel_format(
   1377         content_or_path=path_or_buffer, storage_options=storage_options
   1378     )
   1379     if ext is None:
   1380         raise ValueError(
   1381             "Excel file format cannot be determined, you must specify "
   1382             "an engine manually."
   1383         )

File ~\anaconda3\lib\site-packages\pandas\io\excel\_base.py:1250, in inspect_excel_fo
rmat(content_or_path, storage_options)
   1247 if isinstance(content_or_path, bytes):
   1248     content_or_path = BytesIO(content_or_path)
-> 1250 with get_handle(
   1251     content_or_path, "rb", storage_options=storage_options, is_text=False
   1252 ) as handle:
   1253     stream = handle.handle
   1254     stream.seek(0)

File ~\anaconda3\lib\site-packages\pandas\io\common.py:798, in get_handle(path_or_bu
f, mode, encoding, compression, memory_map, is_text, errors, storage_options)
    789         handle = open(
    790             handle,
    791             ioargs.mode,
   (...)
    794             newline="",
    795         )
    796     else:
    797         # Binary mode
--> 798         handle = open(handle, ioargs.mode)
    799     handles.append(handle)
    801 # Convert BytesIO or file objects passed with an encoding

FileNotFoundError: [Errno 2] No such file or directory: 'loan-train.csv'
```

In [13]:

```
df = pd.read_csv('loan-train.csv')
df.sample(10)
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Input In [13], in <cell line: 1>()
----> 1 df = pd.read_csv('loan-train.csv')
      2 df.sample(10)

File ~\anaconda3\lib\site-packages\pandas\util\_decorators.py:311, in deprecate_nonke
yword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    305 if len(args) > num_allow_args:
    306     warnings.warn(
    307         msg.format(arguments=arguments),
    308         FutureWarning,
    309         stacklevel=stacklevel,
    310     )
--> 311 return func(*args, **kwargs)

File ~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py:680, in read_csv(file
path_or_buffer, sep, delimiter, header, names, index_col, usecols, squeeze, prefix, m
angle_dupe_cols, dtype, engine, converters, true_values, false_values, skipinitialspa
ce, skiprows, skipfooter, nrows, na_values, keep_default_na, na_filter, verbose, skip
_blank_lines, parse_dates, infer_datetime_format, keep_date_col, date_parser, dayfirs
t, cache_dates, iterator, chunksize, compression, thousands, decimal, lineterminator,
quotechar, quoting, doublequote, escapechar, comment, encoding, encoding_errors, dial
ect, error_bad_lines, warn_bad_lines, on_bad_lines, delim_whitespace, low_memory, mem
ory_map, float_precision, storage_options)
    665 kwds_defaults = _refine_defaults_read(
    666     dialect,
    667     delimiter,
   (...)
    676     defaults={"delimiter": ","},
    677 )
    678 kwds.update(kwds_defaults)
--> 680 return _read(filepath_or_buffer, kwds)

File ~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py:575, in _read(filepat
h_or_buffer, kwds)
    572 _validate_names(kwds.get("names", None))
    574 # Create the parser.
--> 575 parser = TextFileReader(filepath_or_buffer, **kwds)
    577 if chunksize or iterator:
    578     return parser

File ~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py:933, in TextFileReade
r.__init__(self, f, engine, **kwds)
    930     self.options["has_index_names"] = kwds["has_index_names"]
    932 self.handles: IOHandles | None = None
--> 933 self._engine = self._make_engine(f, self.engine)

File ~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py:1217, in TextFileRead
er._make_engine(self, f, engine)
   1213     mode = "rb"
   1214 # error: No overload variant of "get_handle" matches argument types
   1215 # "Union[str, PathLike[str], ReadCsvBuffer[bytes], ReadCsvBuffer[str]]"
   1216 # , "str", "bool", "Any", "Any", "Any", "Any", "Any"
-> 1217 self.handles = get_handle(  # type: ignore[call-overload]
   1218     f,
   1219     mode,
   1220     encoding=self.options.get("encoding", None),
   1221     compression=self.options.get("compression", None),
   1222     memory_map=self.options.get("memory_map", False),
   1223         is_text=is_text,
   1224         errors=self.options.get("encoding_errors", "strict"),
   1225         storage_options=self.options.get("storage_options", None),
   1226     )
   1227     assert self.handles is not None
   1228     f = self.handles.handle

File ~\anaconda3\lib\site-packages\pandas\io\common.py:789, in get_handle(path_or_bu
f, mode, encoding, compression, memory_map, is_text, errors, storage_options)
    784 elif isinstance(handle, str):
    785     # Check whether the filename is to be opened in binary mode.
    786     # Binary mode does not support 'encoding' and 'newline'.
    787     if ioargs.encoding and "b" not in ioargs.mode:
    788         # Encoding
--> 789         handle = open(
    790             handle,
    791             ioargs.mode,
    792             encoding=ioargs.encoding,
    793             errors=errors,
    794             newline="",
    795         )
    796     else:
    797         # Binary mode
    798         handle = open(handle, ioargs.mode)

FileNotFoundError: [Errno 2] No such file or directory: 'loan-train.csv'
```

In [14]:

```
df = pd.read_excel('loan-train.csv')
df.sample()
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Input In [14], in <cell line: 1>()
----> 1 df = pd.read_excel('loan-train.csv')
      2 df.sample()

File ~\anaconda3\lib\site-packages\pandas\util\_decorators.py:311, in deprecate_nonke
yword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    305 if len(args) > num_allow_args:
    306     warnings.warn(
    307         msg.format(arguments=arguments),
    308         FutureWarning,
    309         stacklevel=stacklevel,
    310     )
--> 311 return func(*args, **kwargs)

File ~\anaconda3\lib\site-packages\pandas\io\excel\_base.py:457, in read_excel(io, sh
eet_name, header, names, index_col, usecols, squeeze, dtype, engine, converters, true
_values, false_values, skiprows, nrows, na_values, keep_default_na, na_filter, verbos
e, parse_dates, date_parser, thousands, decimal, comment, skipfooter, convert_float,
 mangle_dupe_cols, storage_options)
    455 if not isinstance(io, ExcelFile):
    456     should_close = True
--> 457     io = ExcelFile(io, storage_options=storage_options, engine=engine)
    458 elif engine and engine != io.engine:
    459     raise ValueError(
    460         "Engine should not be specified when passing "
    461         "an ExcelFile - ExcelFile already has the engine set"
    462     )

File ~\anaconda3\lib\site-packages\pandas\io\excel\_base.py:1376, in ExcelFile.__init
__(self, path_or_buffer, engine, storage_options)
   1374     ext = "xls"
   1375 else:
-> 1376     ext = inspect_excel_format(
   1377         content_or_path=path_or_buffer, storage_options=storage_options
   1378     )
   1379     if ext is None:
   1380         raise ValueError(
   1381             "Excel file format cannot be determined, you must specify "
   1382             "an engine manually."
   1383         )

File ~\anaconda3\lib\site-packages\pandas\io\excel\_base.py:1250, in inspect_excel_fo
rmat(content_or_path, storage_options)
   1247 if isinstance(content_or_path, bytes):
   1248     content_or_path = BytesIO(content_or_path)
-> 1250 with get_handle(
   1251     content_or_path, "rb", storage_options=storage_options, is_text=False
   1252 ) as handle:
   1253     stream = handle.handle
   1254     stream.seek(0)

File ~\anaconda3\lib\site-packages\pandas\io\common.py:798, in get_handle(path_or_bu
f, mode, encoding, compression, memory_map, is_text, errors, storage_options)
    789         handle = open(
    790             handle,
    791             ioargs.mode,
   (...)
    794             newline="",
    795         )
    796     else:
    797         # Binary mode
--> 798         handle = open(handle, ioargs.mode)
    799     handles.append(handle)
    801 # Convert BytesIO or file objects passed with an encoding

FileNotFoundError: [Errno 2] No such file or directory: 'loan-train.csv'
```

In [15]:
```
import os
```

In [16]:
```
os.getcwd
```

Out[16]:
```
<function nt.getcwd()>
```

In [17]:
```
os.getcwd()
```

Out[17]:
```
'C:\\Users\\Kalekye'
```

In [18]:
```
os.chdir(C:\\Users\\Kalekye\\Desktop\\pandas)
```

```
  Input In [18]
    os.chdir(C:\\Users\\Kalekye\\Desktop\\pandas)
              ^
SyntaxError: invalid syntax
```

In [19]:
```
os.chdir('C:\\Users\\Kalekye\\Desktop\\pandas')
```

In [20]:
```
os.getcwd()
```

Out[20]:
```
'C:\\Users\\Kalekye\\Desktop\\pandas'
```

In [23]:
```
df = pd.read_csv('loan-train.csv')
df.head(10)
```

Out[23]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplicant
0	LP001002	Male	No	0	Graduate	No	5849	
1	LP002840	Female	No	0	Graduate	No	2378	
2	LP001030	Male	Yes	2	Graduate	No	1299	
3	LP001325	Male	No	0	Not Graduate	No	3620	
4	LP001482	Male	Yes	0	Graduate	Yes	3459	
5	LP002792	Male	Yes	1	Graduate	No	5468	
6	LP001518	Male	Yes	1	Graduate	No	1538	
7	LP001888	Female	No	0	Graduate	No	3237	
8	LP001086	Male	No	0	Not Graduate	No	1442	
9	LP002894	Female	Yes	0	Graduate	No	3166	

In [25]:

```
cat_features = (i for i in df.columns if df[i].dtype =='O')
cat_features
```

Out[25]:

```
<generator object <genexpr> at 0x000001E280312A50>
```

In [28]:

```
cat_features = [i for i in df.columns if df[i].dtype =='O']
cat_features.remove('Loan_ID')
cat_features.remove('Loan_Status')
cat_features
```

Out[28]:

```
['Gender',
 'Married',
 'Dependents',
 'Education',
 'Self_Employed',
 'Property_Area']
```

In [29]:

```
num_features = [i for i in df.columns if df[i].dtype !='O']
num_features
```

Out[29]:

```
['ApplicantIncome',
 'CoapplicantIncome',
 'LoanAmount',
 'Loan_Amount_Term',
 'Credit_History']
```

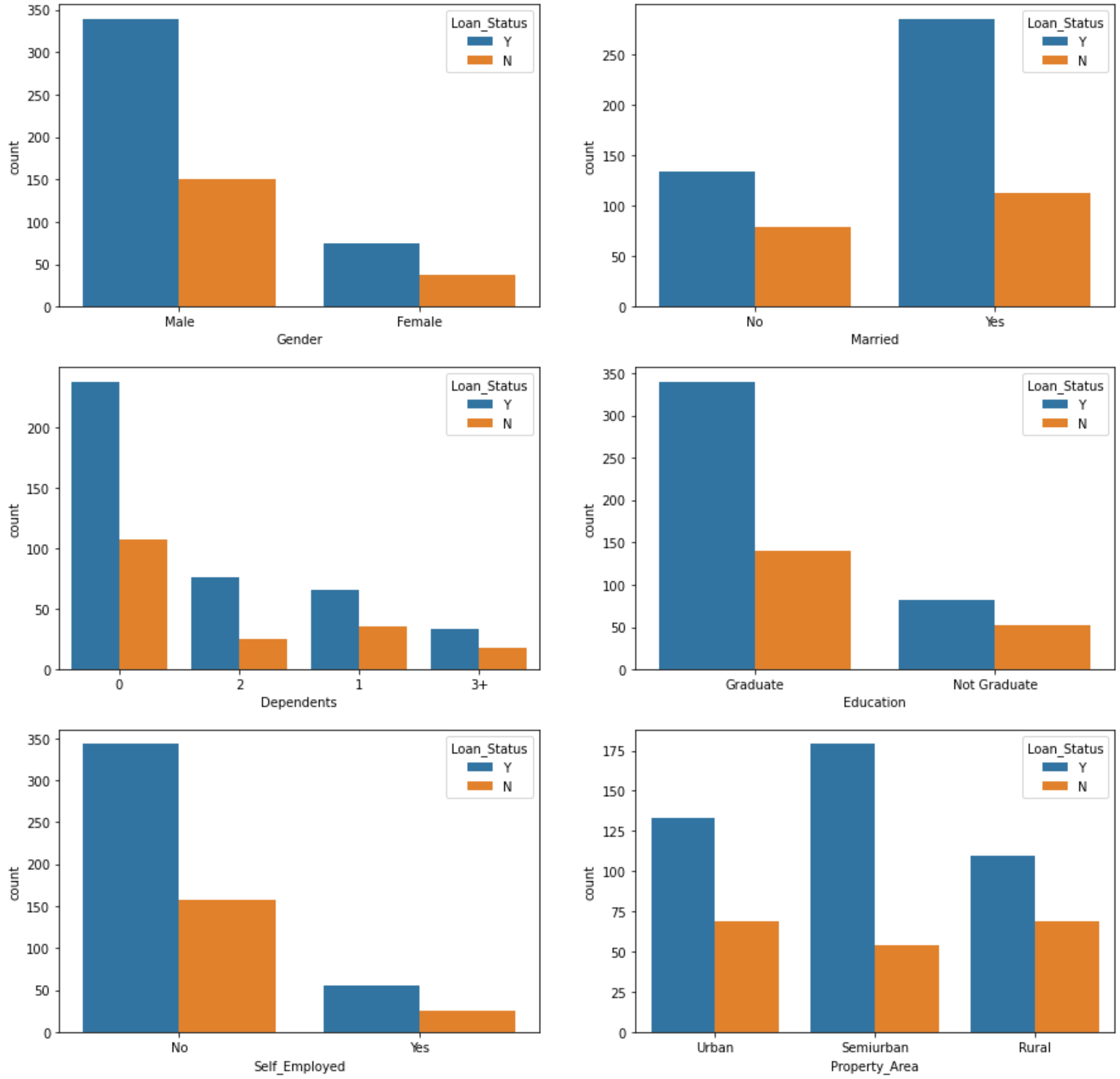
In [30]:

```
fig, ax = plt.subplots(3,2 figsize = (15,15))
axs = ax.ravel()
for i, features in enumerate(cat_features):
    sns.countplot(x = features, hue= 'Loan_Status', ax = axs[i], data = df)
```

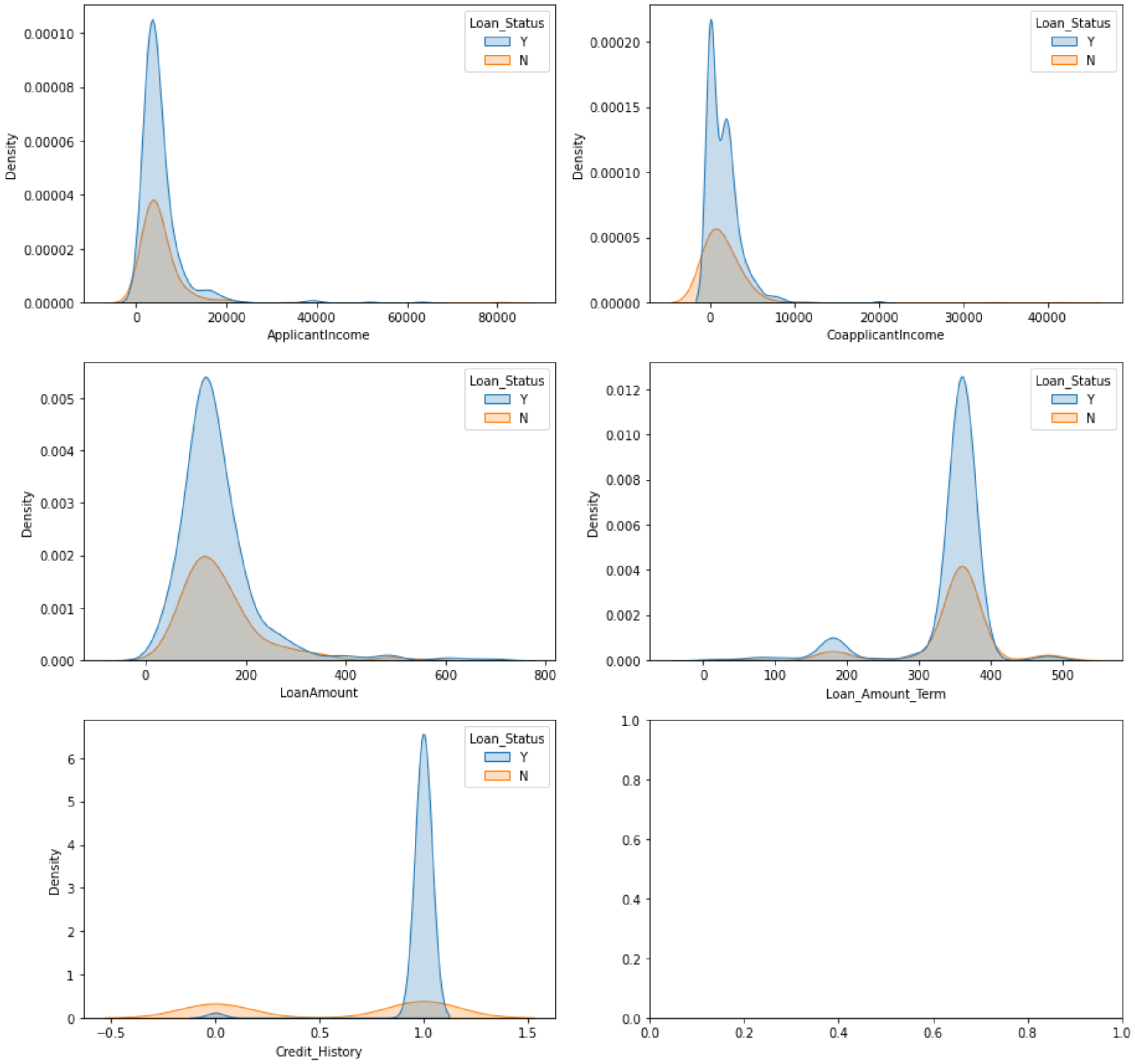
```
  Input In [30]
    fig, ax = plt.subplots(3,2 figsize = (15,15))
                               ^
SyntaxError: invalid syntax
```

In [31]:

```
fig, ax = plt.subplots(3,2, figsize = (15,15))
axs = ax.ravel()
for i, features in enumerate(cat_features):
    sns.countplot(x = features, hue= 'Loan_Status', ax = axs[i], data = df)
```

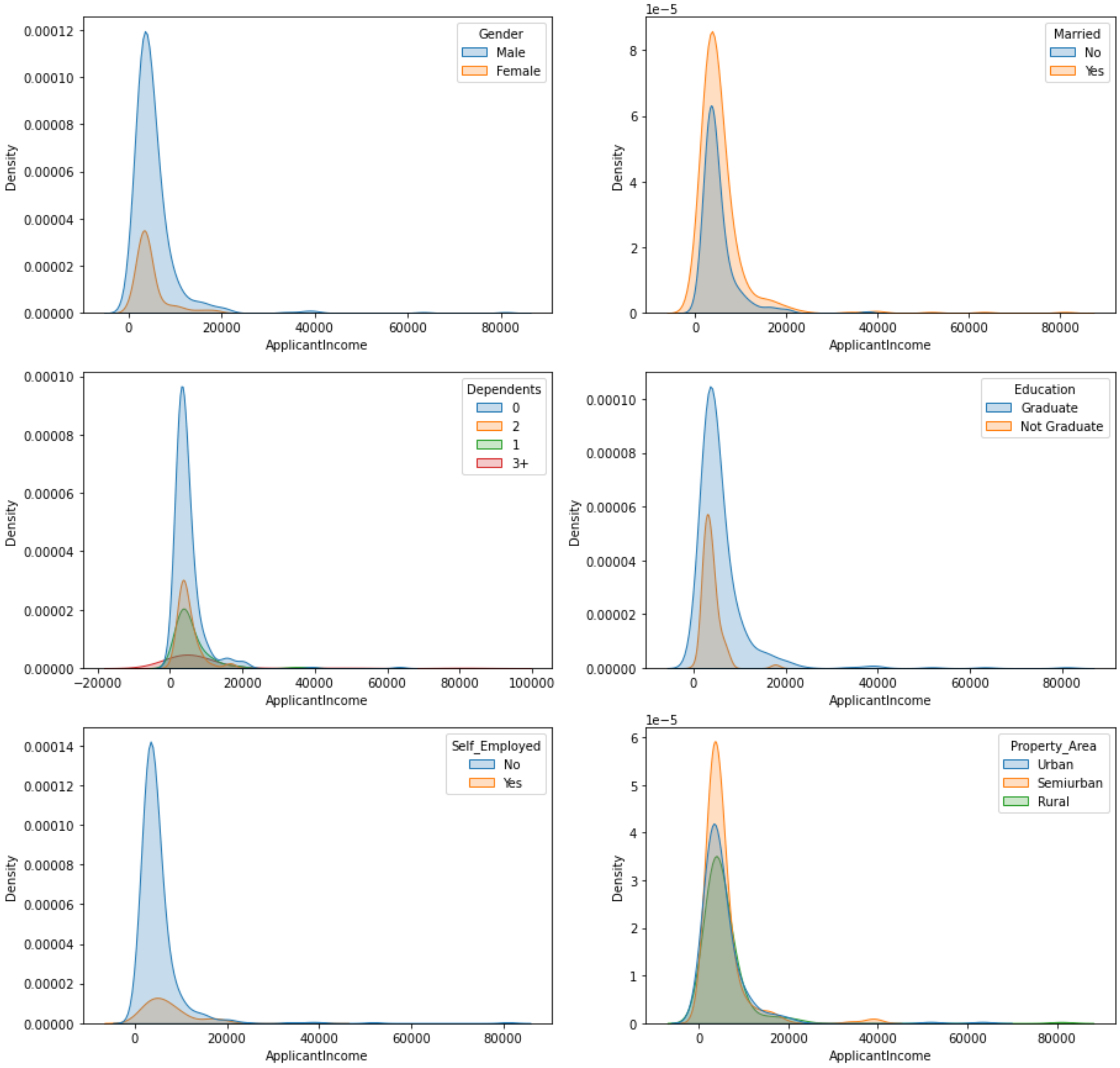


In [34]:
```
fig, ax = plt.subplots(3,2, figsize = (15,15))
axs = ax.ravel()
for i, features in enumerate(num_features):
    sns.kdeplot(x = features, hue= 'Loan_Status', ax = axs[i], data = df, fill = True)
```



In [37]:

```
fig, ax = plt.subplots(3,2, figsize = (15,15))
axs = ax.ravel()
for i, features in enumerate(cat_features):
    sns.kdeplot(x = 'ApplicantIncome', hue= features, ax = axs[i], data = df, fill = T
```


Gender
Male
Female
Density
0.00000
0.00002
0.00004
0.00006
0.00008
0.00010
0.00012
0
20000
40000
60000
80000
ApplicantIncome
1e−5
Married
No
Yes
Density
0
2
4
6
8
ApplicantIncome
Dependents
0
2
1
3+
Density
0.00000
0.00002
0.00004
0.00006
0.00008
0.00010
−20000
0
20000
40000
60000
80000
100000
ApplicantIncome
Education
Graduate
Not Graduate
Density
0.00000
0.00002
0.00004
0.00006
0.00008
0.00010
ApplicantIncome
Self_Employed
No
Yes
Density
0.00000
0.00002
0.00004
0.00006
0.00008
0.00010
0.00012
0.00014
ApplicantIncome
1e−5
Property_Area
Urban
Semiurban
Rural
Density
0
1
2
3
4
5
6
ApplicantIncome

```
In [ ]:
```