```
In [1]:
import re
import os

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

```
In [2]:
# File Paths
data_path = "combined_results.csv"
```

```
In [3]:
# reading in csv and converting to dataframes
data = pd.read_csv(data_path, error_bad_lines=False,low_memory=False)
```

```
In [4]:
data
```

Out[4]:

	Amount_Requested	Risk_Score	debt_to_income_ratio	State	Emp_length	Purpose	Loan_S
0	5000.0	669.0	21.80	OK	8	other	Acc
1	15000.0	704.0	18.29	FL	2	debt_consolidation	Acc
2	11200.0	669.0	43.97	NH	0	medical	Acc
3	25000.0	669.0	12.89	AL	10	debt_consolidation	Acc
4	3000.0	764.0	0.58	WA	9	major_purchase	Acc
...	...	...	...	...	...	...	
1081515	3000.0	535.0	1.54	CT	5	Medical expenses	Re
1081516	6000.0	580.0	5.55	TX	0	Car financing	Re
1081517	3000.0	561.0	0.00	OR	0	Debt consolidation	Re
1081518	2500.0	573.0	8.55	NY	0	Other	Re
1081519	2500.0	567.0	0.00	AL	2	Car financing	Re

1081520 rows × 7 columns

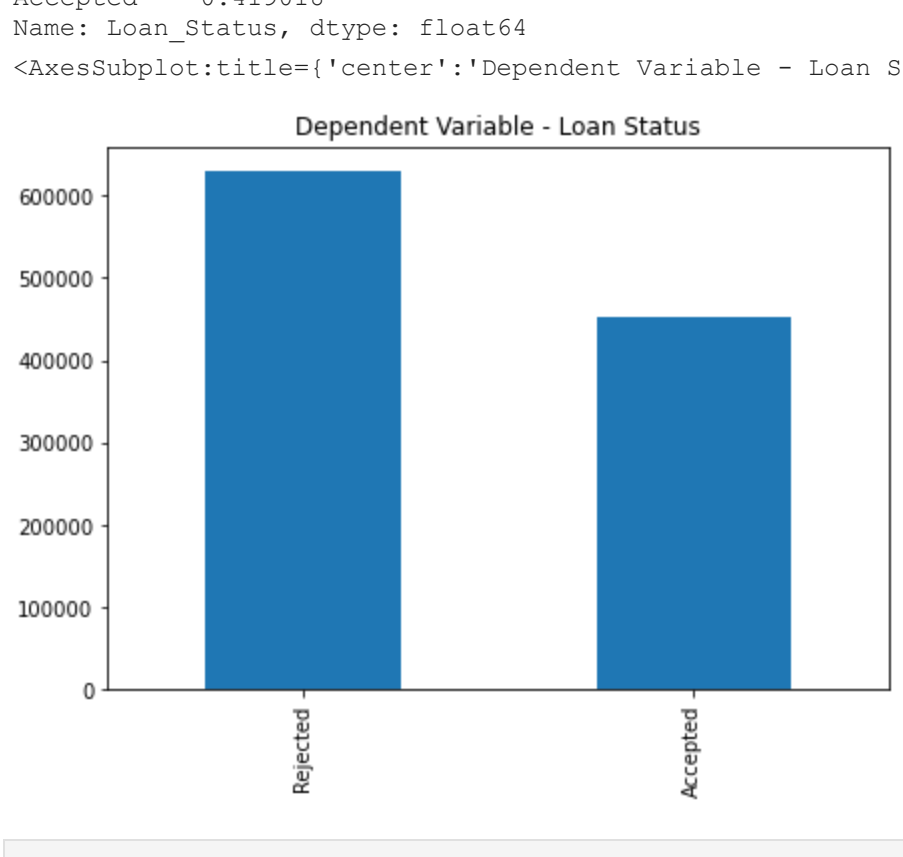
Cont'd Exploratory Data analysis

Univariate exploration

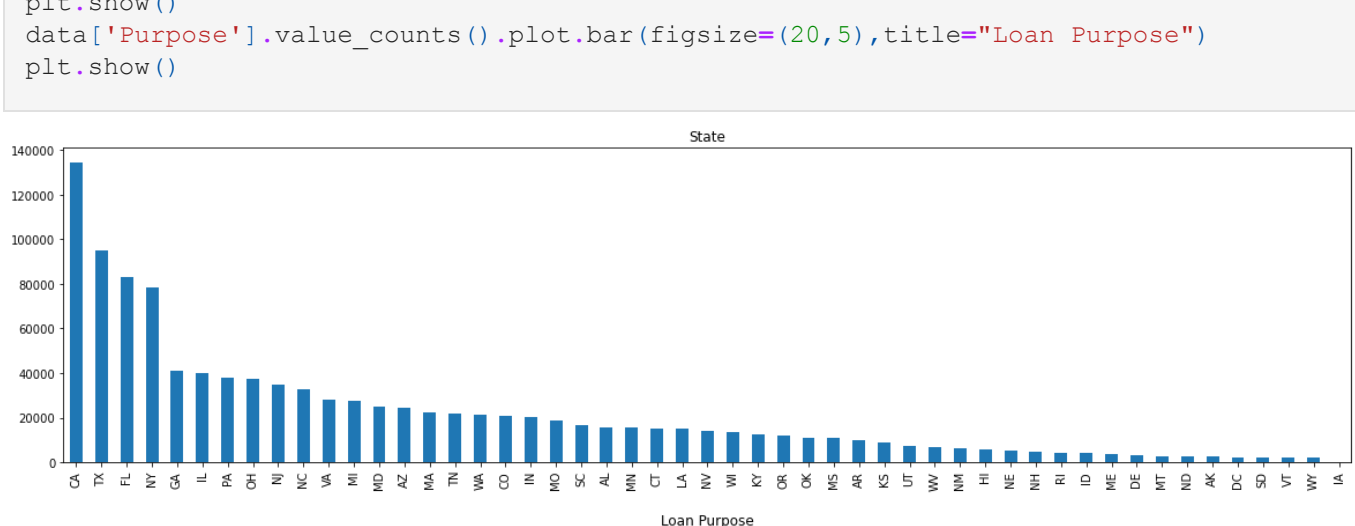
```
In [5]:
# Understanding the split of our Accept/Reject - Our Target columns
print(data['Loan_Status'].value_counts(normalize=True))

data['Loan_Status'].value_counts().plot.bar(figsize=(7,5),title="Dependent Variable -
```

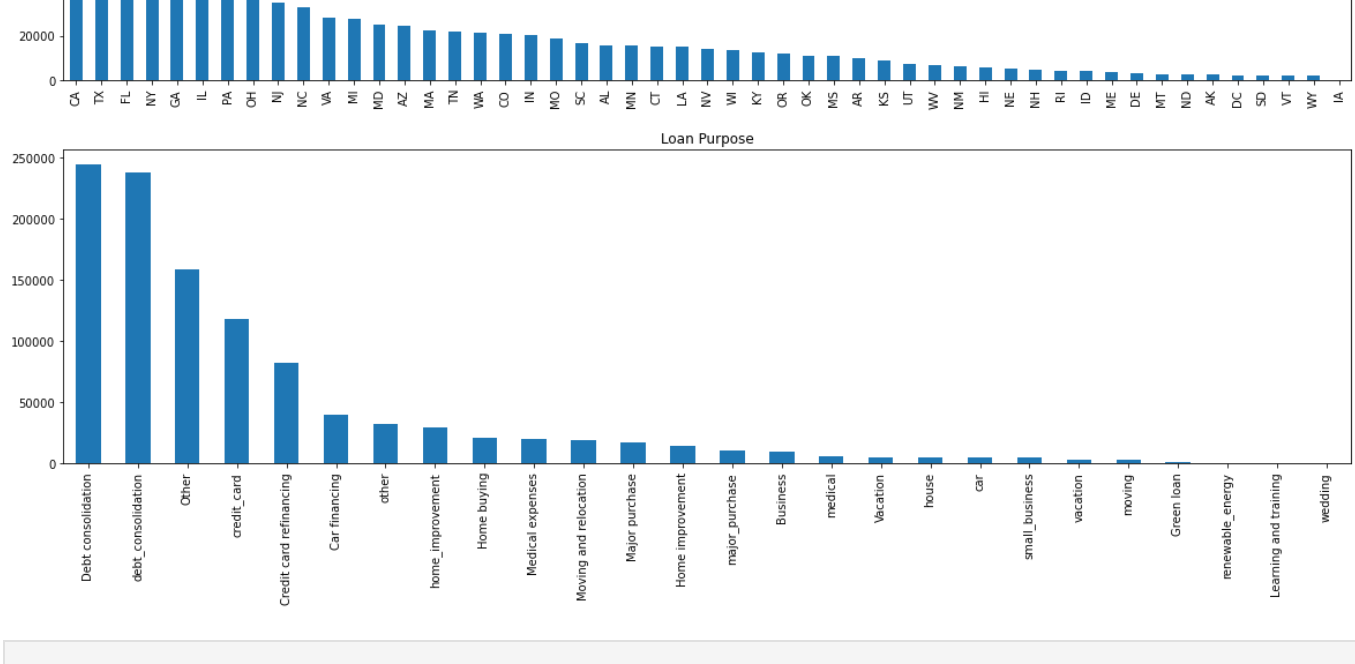
```
Rejected    0.580982
Accepted    0.419018
Name: Loan_Status, dtype: float64
```

```
Out[5]: <AxesSubplot:title={'center':'Dependent Variable - Loan Status'}>
```



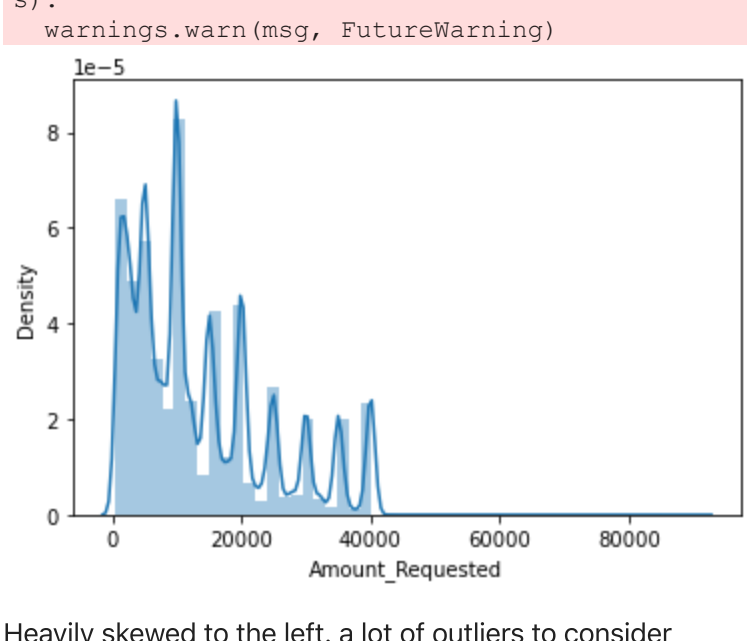

```
In [6]:
# understanding split of all categorical Columns
data['State'].value_counts().plot.bar(figsize=(20,5),title="State")
plt.show()
data['Purpose'].value_counts().plot.bar(figsize=(20,5),title="Loan Purpose")
plt.show()
```






```
In [7]:
# understanding split of all Numeric Columns
sns.distplot(data['Amount_Requested'])
plt.show()
```

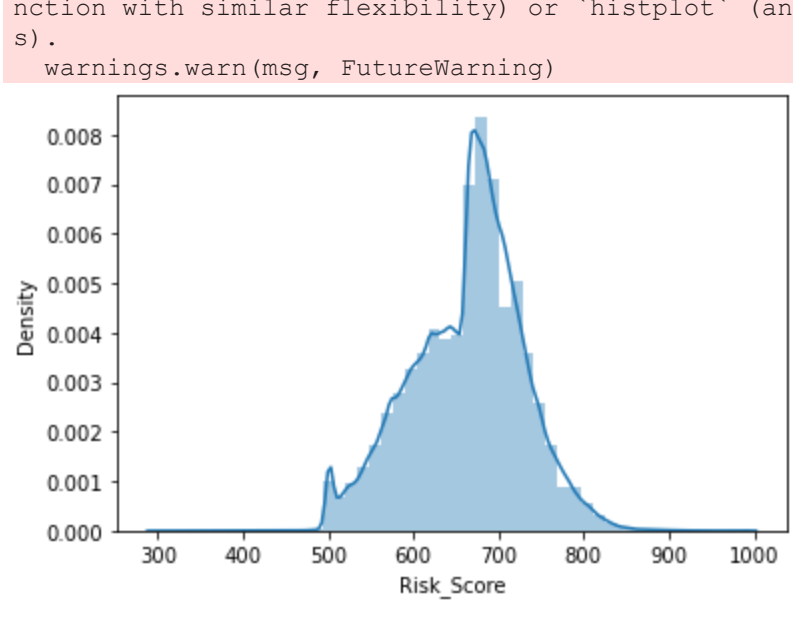
```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Heavily skewed to the left. a lot of outliers to consider

```
In [8]:
# understanding split of all Numeric Columns
sns.distplot(data['Risk_Score'])
plt.show()
```

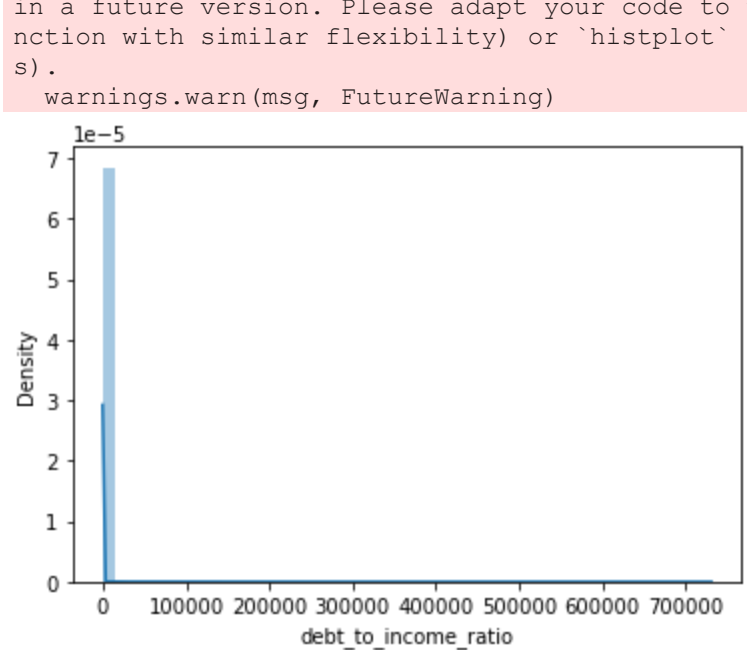
```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Distribution is good

```
In [9]:
# understanding split of all Numeric Columns
sns.distplot(data['debt_to_income_ratio'])
plt.show()

print(data['debt_to_income_ratio'].max())
print('---------------')
print(data['debt_to_income_ratio'].min())
print('---------------')
print(data[['debt_to_income_ratio','Loan_Status']].sort_values(by='debt_to_income_rat
```

```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```




```
730426.5
---------------
0.0
---------------
        debt_to_income_ratio Loan_Status
1081519                  0.0    Rejected
72018                    0.0    Accepted
912669                   0.0    Rejected
912668                   0.0    Rejected
912667                   0.0    Rejected
...                      ...         ...
752158              167650.0    Rejected
922204              183650.0    Rejected
810079              192200.0    Rejected
811546              196100.0    Rejected
783878              730426.5    Rejected

[1081520 rows x 2 columns]
```

```
In [10]:
# df[df.b > 10]
dti_values_over100 = data[data.debt_to_income_ratio > 100]
```

```
In [11]:
print(dti_values_over100.info())
dti_values_over100.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22081 entries, 493 to 1081423
Data columns (total 7 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Amount_Requested      22081 non-null  float64
 1   Risk_Score            22081 non-null  float64
 2   debt_to_income_ratio  22081 non-null  float64
 3   State                 22081 non-null  object
 4   Emp_length            22081 non-null  int64
 5   Purpose               22081 non-null  object
 6   Loan_Status           22081 non-null  object
dtypes: float64(3), int64(1), object(3)
memory usage: 1.3+ MB
None
```

Out[11]:

	Amount_Requested	Risk_Score	debt_to_income_ratio	State	Emp_length	Purpose	Loan_Stat
493	28000.0	704.0	112.20	NV	0	debt_consolidation	Accept
643	40000.0	709.0	101.46	OH	4	debt_consolidation	Accept
1229	10000.0	714.0	114.27	TX	9	debt_consolidation	Accept
1683	9000.0	684.0	106.66	IL	4	credit_card	Accept
1879	35000.0	714.0	102.47	UT	0	debt_consolidation	Accept

```
In [12]:
dti_values_over100['Loan_Status'].value_counts()
```

```
Out[12]:
Rejected    21193
Accepted      888
Name: Loan_Status, dtype: int64
```

It appears that a lot of the DTI values Over 100 (95%+) are those who were rejected. Due to the nature that being a severe outlier, I think it would be best to reject anyone whose DTI is over 100.

Let's remove them.

```
In [13]:
clean_dti_data = data[data.debt_to_income_ratio < 100]
```

```
In [14]:
clean_dti_data
```

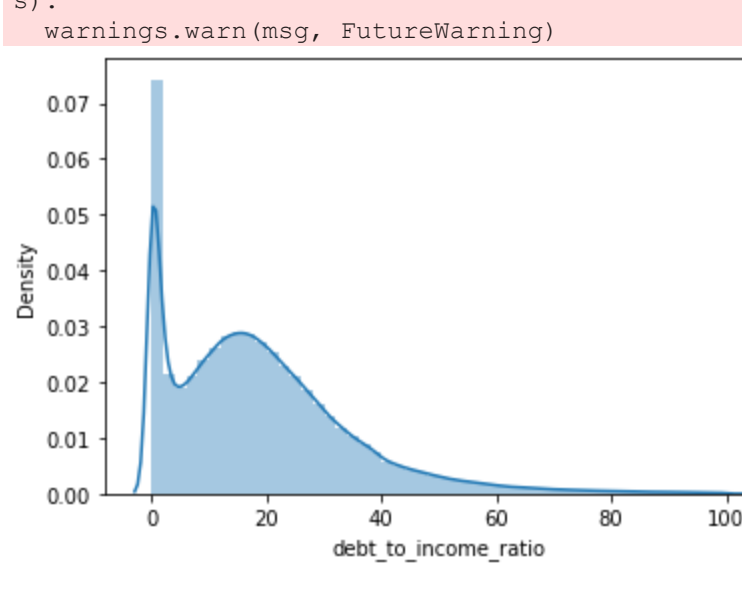
Out[14]:

	Amount_Requested	Risk_Score	debt_to_income_ratio	State	Emp_length	Purpose	Loan_S
0	5000.0	669.0	21.80	OK	8	other	Acc
1	15000.0	704.0	18.29	FL	2	debt_consolidation	Acc
2	11200.0	669.0	43.97	NH	0	medical	Acc
3	25000.0	669.0	12.89	AL	10	debt_consolidation	Acc
4	3000.0	764.0	0.58	WA	9	major_purchase	Acc
...	...	...	...	...	...	...	
1081515	3000.0	535.0	1.54	CT	5	Medical expenses	Re
1081516	6000.0	580.0	5.55	TX	0	Car financing	Re
1081517	3000.0	561.0	0.00	OR	0	Debt consolidation	Re
1081518	2500.0	573.0	8.55	NY	0	Other	Re
1081519	2500.0	567.0	0.00	AL	2	Car financing	Re

1034236 rows × 7 columns

```
In [15]:
sns.distplot(clean_dti_data['debt_to_income_ratio'])
plt.show()
```

```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

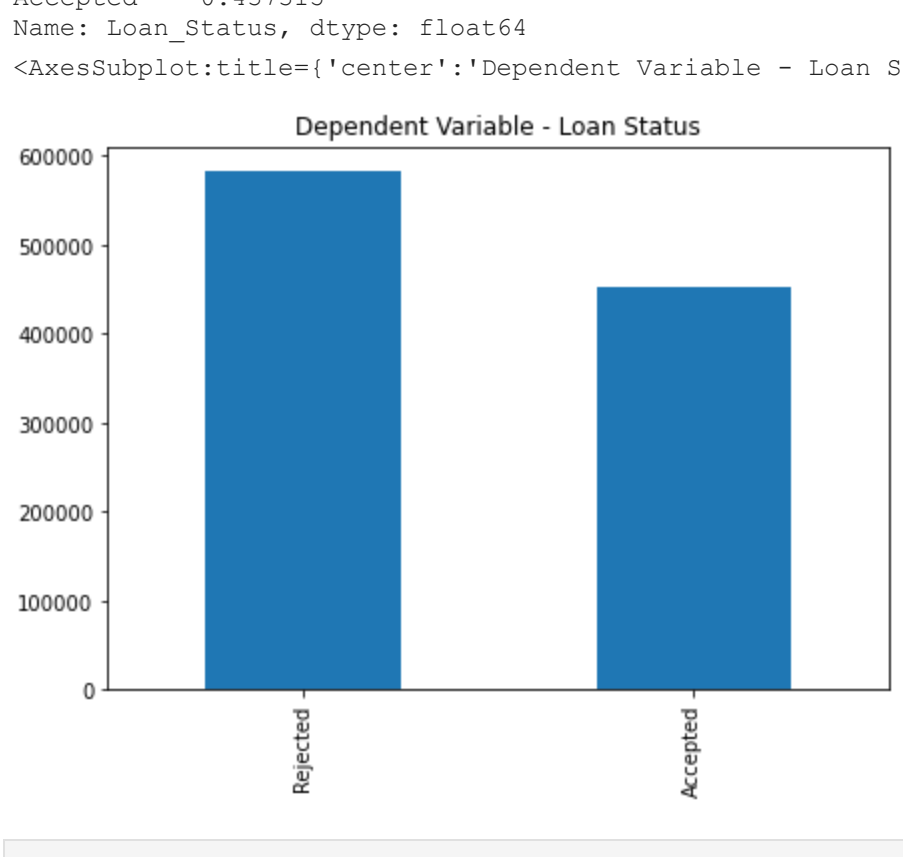


replotting with cleaned data

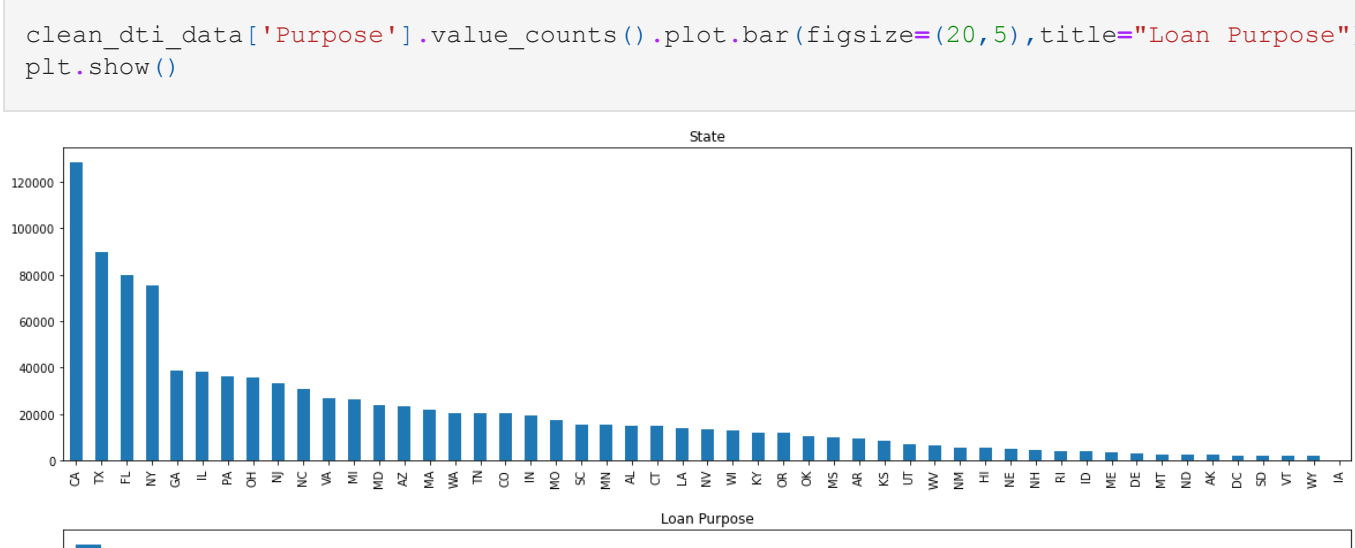
```
In [16]:
print(clean_dti_data['Loan_Status'].value_counts(normalize=True))

clean_dti_data['Loan_Status'].value_counts().plot.bar(figsize=(7,5),title="Dependent V
```

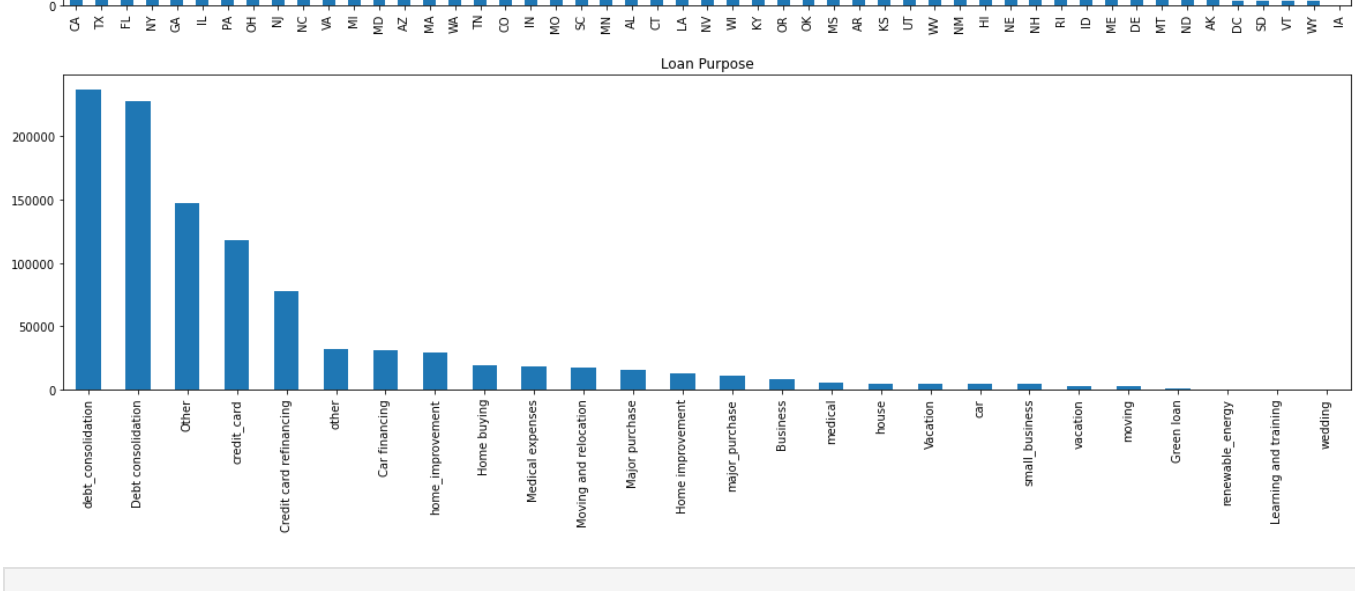
```
Rejected    0.562685
Accepted    0.437315
Name: Loan_Status, dtype: float64
```

```
Out[16]: <AxesSubplot:title={'center':'Dependent Variable - Loan Status'}>
```



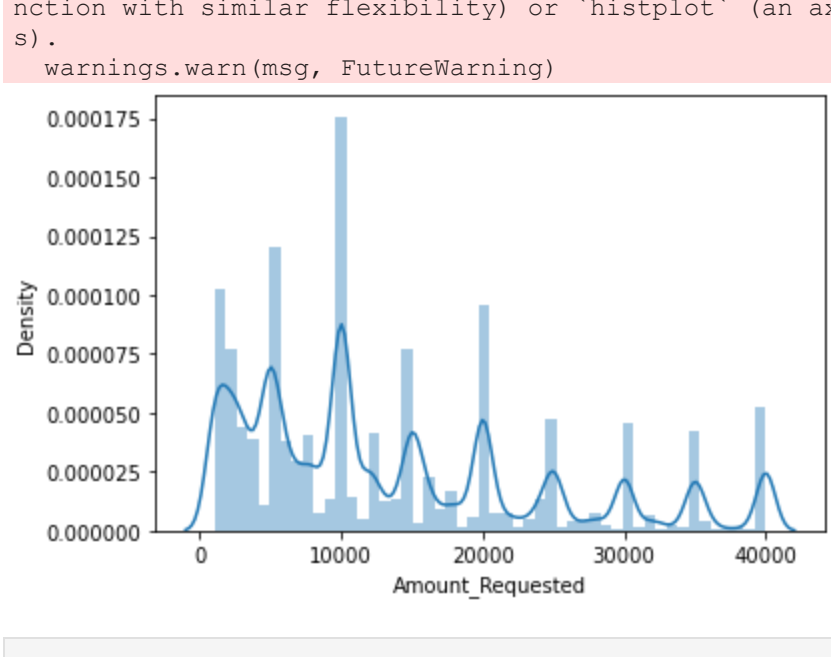

```
In [17]:
# understanding split of all categorical Columns in cleaned data

clean_dti_data['State'].value_counts().plot.bar(figsize=(20,5),title="State")
plt.show()

clean_dti_data['Purpose'].value_counts().plot.bar(figsize=(20,5),title="Loan Purpose")
plt.show()
```





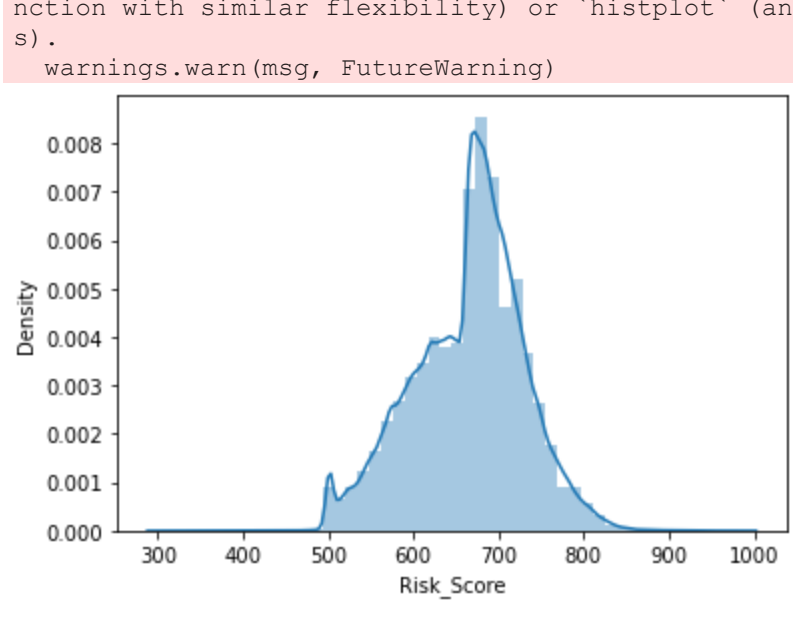

```
In [18]:
# understanding split of all Numeric Columns in cleaned data
sns.distplot(clean_dti_data['Amount_Requested'])
plt.show()
```

```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```




```
In [19]:
# understanding split of all Numeric Columns
sns.distplot(clean_dti_data['Risk_Score'])
plt.show()
```

```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Majority of the data remains relatively unchanged after cleaning out DTI

However, I need to recategorize purpose of loan

```
In [20]:
# purpose of Loans need to be recatogorized
# clean_dti_data['Purpose'].value_counts()
```

```
In [21]:
clean_dti_data.loc[(clean_dti_data.Purpose == 'debt_consolidation'),'Purpose']='Debt c
clean_dti_data.loc[(clean_dti_data.Purpose == 'other'),'Purpose']='Other'
clean_dti_data.loc[(clean_dti_data.Purpose == 'car'),'Purpose']='Car financing'
clean_dti_data.loc[(clean_dti_data.Purpose == 'home_improvement'),'Purpose']='Home imp
clean_dti_data.loc[(clean_dti_data.Purpose == 'moving'),'Purpose']='Moving and relocat
clean_dti_data.loc[(clean_dti_data.Purpose == 'major_purchase'),'Purpose']='Major purc
clean_dti_data.loc[(clean_dti_data.Purpose == 'medical'),'Purpose']='Medical expenses
clean_dti_data.loc[(clean_dti_data.Purpose == 'house'),'Purpose']='Home buying'
clean_dti_data.loc[(clean_dti_data.Purpose == 'vacation'),'Purpose']='Vacation'
clean_dti_data.loc[(clean_dti_data.Purpose == 'small_business'),'Purpose']='Business'
clean_dti_data.loc[(clean_dti_data.Purpose == 'renewable_energy'),'Purpose']='Green lo
```

```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\pandas\core\indexing.py:1720: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self._setitem_single_column(loc, value, pi)
```

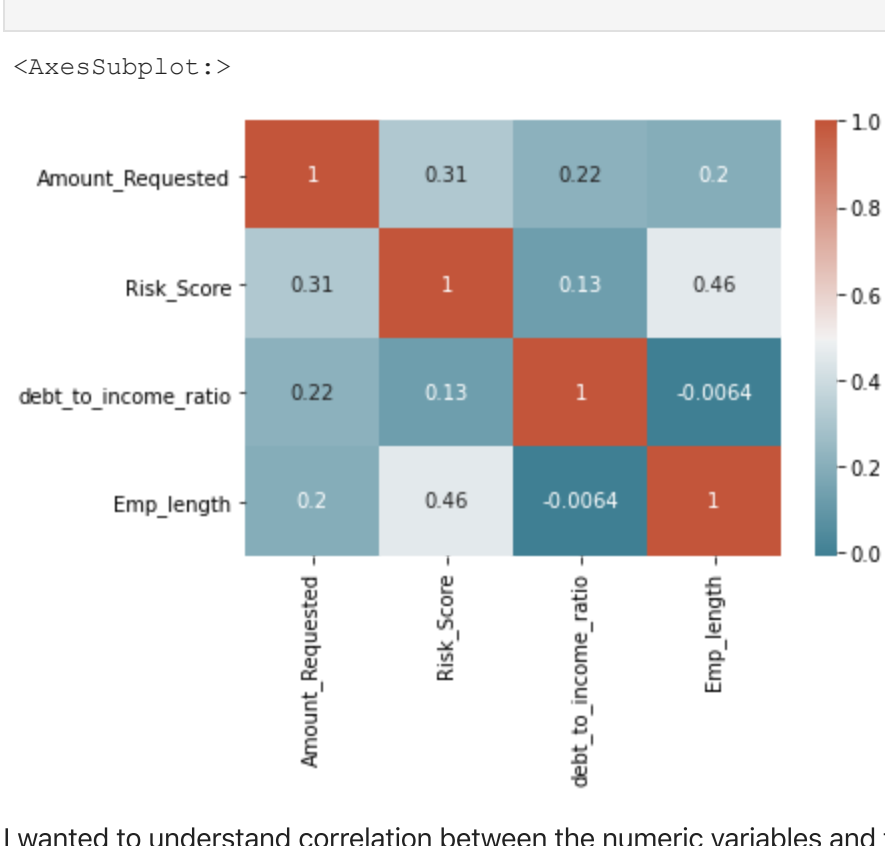
```
In [22]:
# cleaned Purpose column
clean_dti_data['Purpose'].value_counts()
```

```
Out[22]:
Debt consolidation        463923
Other                     179281
credit_card               118031
Credit card refinancing    77889
Home improvement           42409
Car financing              35879
Major purchase             26423
Medical expenses           24098
Home buying                23977
Moving and relocation      20437
Business                   13010
Vacation                    7780
Green loan                  1060
Learning and training         34
wedding                        5
Name: Purpose, dtype: int64
```

Bivariate exploration

```
In [23]:
# calculate correlation matrix
corr = clean_dti_data.corr()# plot the heatmap
sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns, annot=True, cmap
```

```
Out[23]: <AxesSubplot:>
```



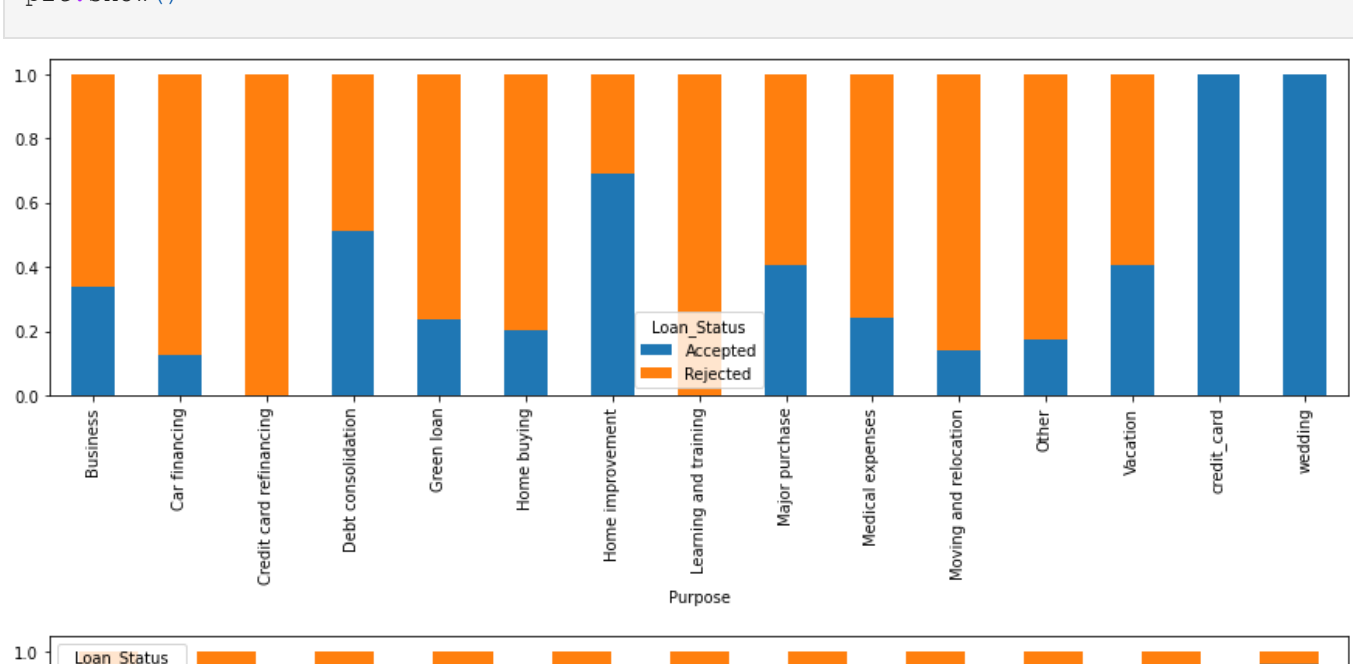
I wanted to understand correlation between the numeric variables and to see if the negative/positive correlation makes sense

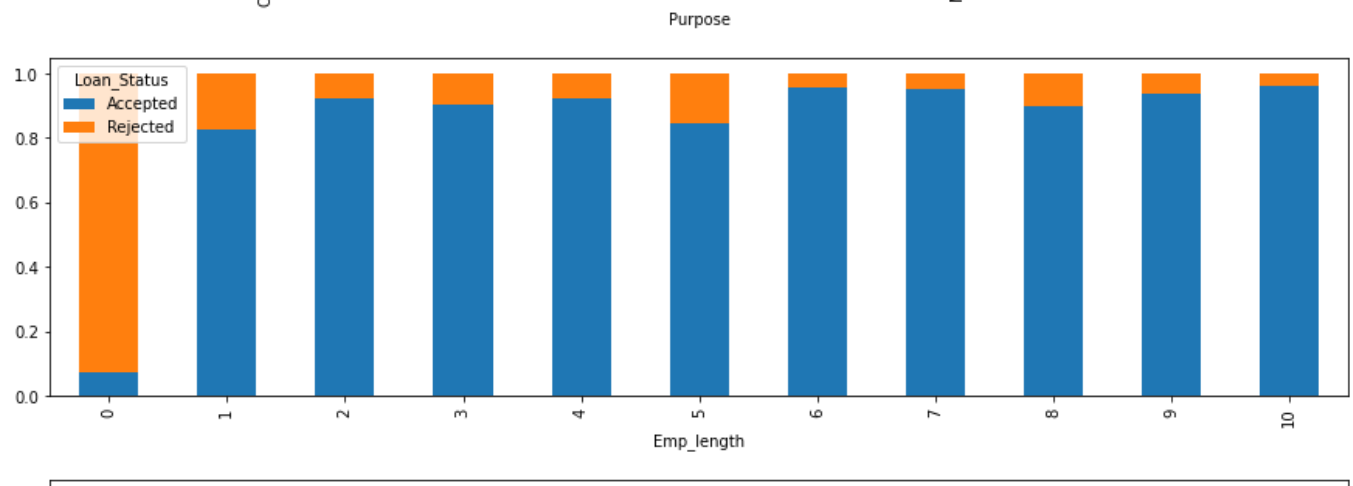
```
In [24]:
# from pandas.plotting import scatter_matrix

# attributes = ['Amount_Requested','Risk_Score','debt_to_income_ratio']
# scatter_matrix(clean_dti_data[attributes], figsize=(12.8))
```

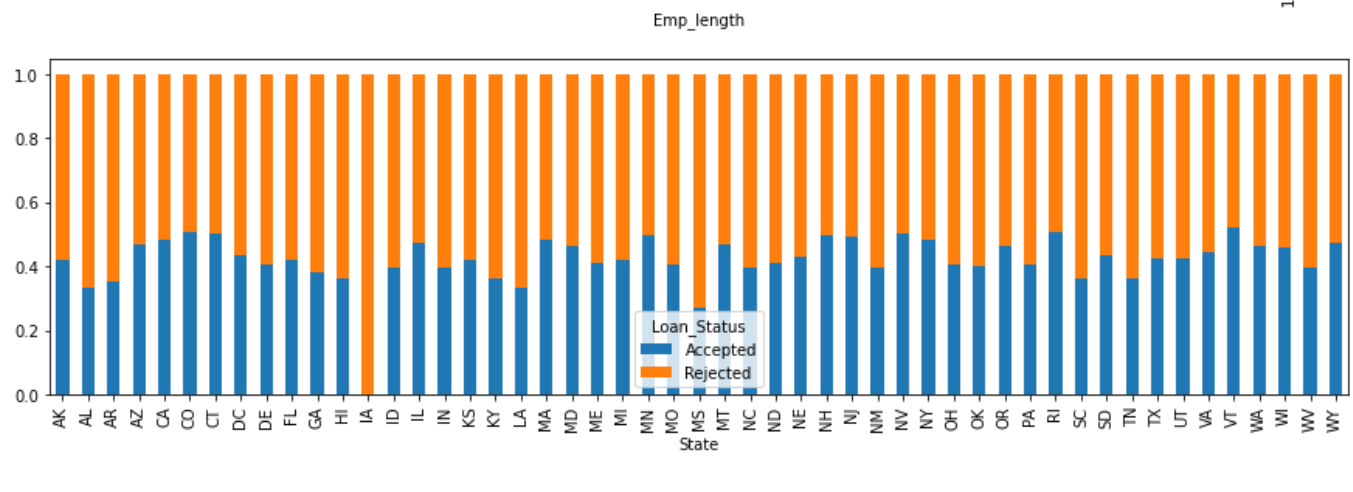
```
In [25]:
clean_dti_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1034236 entries, 0 to 1081519
Data columns (total 7 columns):
 #   Column                Non-Null Count    Dtype
---  ------                --------------    -----
 0   Amount_Requested      1034236 non-null  float64
 1   Risk_Score            1034236 non-null  float64
 2   debt_to_income_ratio  1034236 non-null  float64
 3   State                 1034236 non-null  object
 4   Emp_length            1034236 non-null  int64
 5   Purpose               1034236 non-null  object
 6   Loan_Status           1034236 non-null  object
dtypes: float64(3), int64(1), object(3)
memory usage: 63.1+ MB
```

```
In [26]:
# Splitting all the categorical variables into subsets of the data to see how they in
Loan_purpose = pd.crosstab(clean_dti_data['Purpose'],clean_dti_data['Loan_Status'])
Emp_length = pd.crosstab(clean_dti_data['Emp_length'],clean_dti_data['Loan_Status'])
State = pd.crosstab(clean_dti_data['State'],clean_dti_data['Loan_Status'])

Loan_purpose.div(Loan_purpose.sum(1).astype(float), axis=0).plot(kind='bar',stacked=T
plt.show()
Emp_length.div(Emp_length.sum(1).astype(float), axis=0).plot(kind='bar',stacked=True,
plt.show()
State.div(State.sum(1).astype(float), axis=0).plot(kind='bar',stacked=True,figsize=(1
plt.show()
```

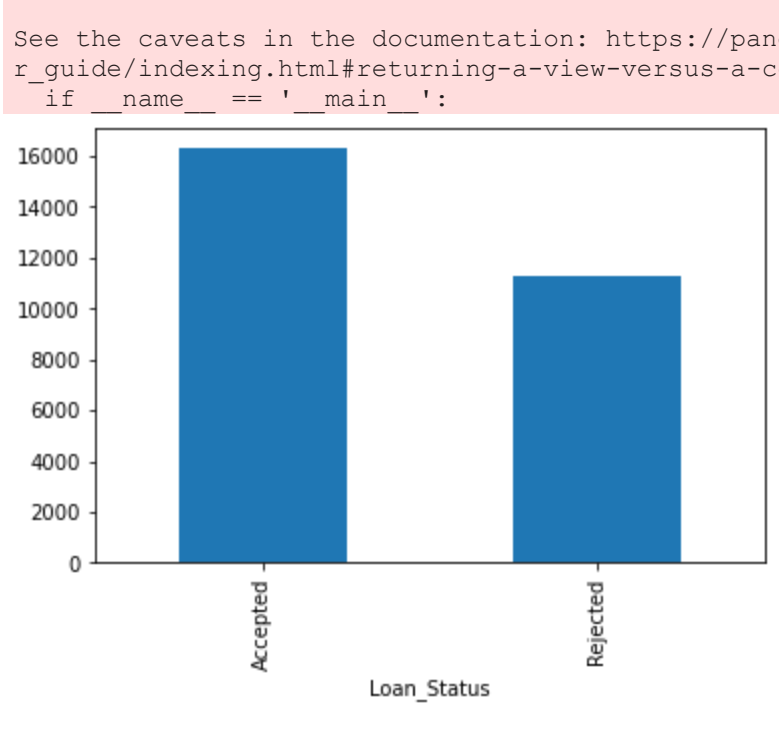


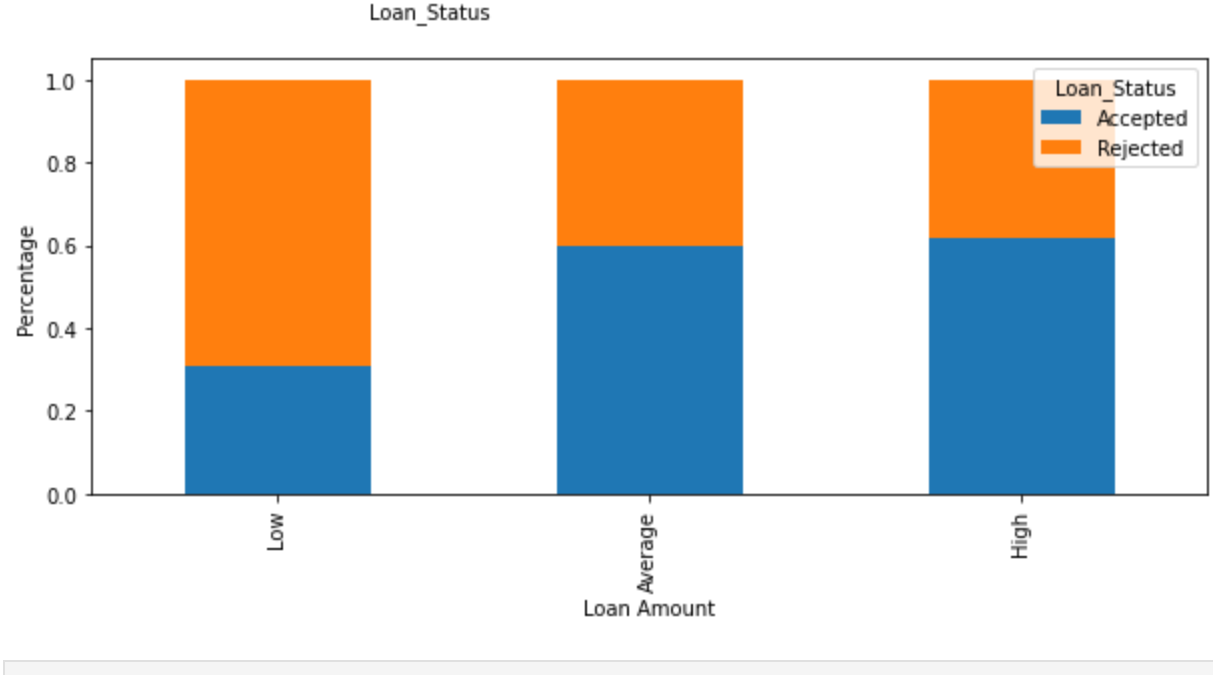





```
In [27]:
# Plot Numeric variables vs target

clean_dti_data.groupby('Loan_Status')['Amount_Requested'].mean().plot.bar()


bins=[0,10000,20000,30000]
group=['Low','Average','High']

clean_dti_data['Amount_Requested_bins']=pd.cut(clean_dti_data['Amount_Requested'],bin

Amount_Requested_bins=pd.crosstab(clean_dti_data['Amount_Requested_bins'],clean_dti_da
Amount_Requested_bins.div(Amount_Requested_bins.sum(1).astype(float), axis=0).plot(ki
plt.xlabel('Loan Amount')
P=plt.ylabel('Percentage')
```

```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\ipykernel_launcher.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  if __name__ == '__main__':
```

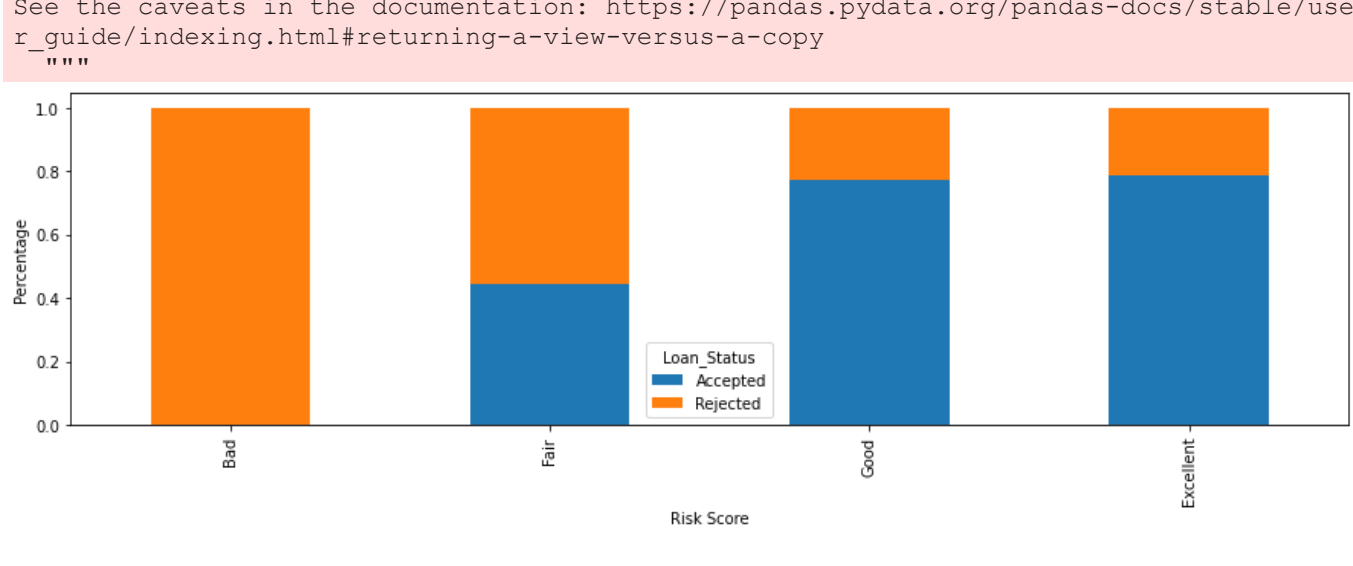





```
In [ ]:
```

```
In [28]:
# plot risk scores
bins=[0,629,689,719,850]
group=['Bad','Fair','Good','Excellent']

clean_dti_data['Risk_Score_bins']=pd.cut(clean_dti_data['Risk_Score'],bins,labels=grou

Risk_Score_bins=pd.crosstab(clean_dti_data['Risk_Score_bins'],clean_dti_data['Loan_Sta
Risk_Score_bins.div(Risk_Score_bins.sum(1).astype(float), axis=0).plot(kind='bar',stac
plt.xlabel('Risk Score')
P=plt.ylabel('Percentage')
```

```
C:\Users\pamponsah\AppData\Local\Continuum\anaconda3\lib\site-packages\ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """
```




```
In [29]:
# lets redo the correlation, but this time let's turn the loan Status into a binary v

clean_dti_data = clean_dti_data.drop(['Risk_Score_bins','Amount_Requested_bins'], axi
```

Create a final table to work with

```
In [30]:
model_data = clean_dti_data.copy()
model_data
```

Out[30]:

	Amount_Requested	Risk_Score	debt_to_income_ratio	State	Emp_length	Purpose	Loan_Status
0	5000.0	669.0	21.80	OK	8	Other	Accepted
1	15000.0	704.0	18.29	FL	2	Debt consolidation	Accepted
2	11200.0	669.0	43.97	NH	0	Medical expenses	Accepted
3	25000.0	669.0	12.89	AL	10	Debt consolidation	Accepted
4	3000.0	764.0	0.58	WA	9	Major purchase	Accepted
...	...	...	...	...	...	...	...
1081515	3000.0	535.0	1.54	CT	5	Medical expenses	Rejected
1081516	6000.0	580.0	5.55	TX	0	Car financing	Rejected
1081517	3000.0	561.0	0.00	OR	0	Debt consolidation	Rejected
1081518	2500.0	573.0	8.55	NY	0	Other	Rejected
1081519	2500.0	567.0	0.00	AL	2	Car financing	Rejected

1034236 rows × 7 columns

```
In [31]:
# Turning Loan status to a binary variable
model_data['Loan_Status'].replace('Accepted',1,inplace=True)
model_data['Loan_Status'].replace('Rejected',0,inplace=True)
```

```
In [32]:
new_corr = model_data.corr()
f, ax = plt.subplots(figsize=(9,6))
sns.heatmap(new_corr,vmax=.8,square=True,cmap='BuPu', annot = True)
```

```
Out[32]: <AxesSubplot:>
```

It looks like all correlation are correctly assumed

Best Correlations:

Employment Length

Risk Scoore

Debt to Income Ration

Model building

```
In [33]:
# identify Dependent and independent variable
y = model_data.Loan_Status
X = model_data.drop('Loan_Status',axis=1)

# might decide to drop state
```

```
In [34]:
X = pd.get_dummies(X)
```

```
In [35]:
X
```

Out[35]:

	Amount_Requested	Risk_Score	debt_to_income_ratio	Emp_length	State_AK	State_AL	State_AR
0	5000.0	669.0	21.80	8	0	0	0
1	15000.0	704.0	18.29	2	0	0	0
2	11200.0	669.0	43.97	0	0	0	0
3	25000.0	669.0	12.89	10	0	1	0
4	3000.0	764.0	0.58	9	0	0	0
...	...	...	...	...	...	...	...
1081515	3000.0	535.0	1.54	5	0	0	0
1081516	6000.0	580.0	5.55	0	0	0	0
1081517	3000.0	561.0	0.00	0	0	0	0
1081518	2500.0	573.0	8.55	0	0	0	0
1081519	2500.0	567.0	0.00	2	0	1	0

1034236 rows × 70 columns

```
In [36]:
# Split to test/train split and scale
from sklearn.model_selection import train_test_split

X_train,X_test,y_train,y_test = train_test_split(X,y, random_state=42)
```

Stochastic Gradient Descent (SGD) classifier

This classifier has the advantage of being capable of handling very large datasets efficiently. This is in part because SGD deals with training instances independently.

Let's create an SGDClassifier and train it on the whole training set:

```
In [37]:
from sklearn.linear_model import SGDClassifier

sgd_clf = SGDClassifier(random_state=42)

sgd_clf.fit(X_train, y_train)
```

```
Out[37]: SGDClassifier(random_state=42)
```

```
In [101…
# for testing purposes - Lets see how good/bad the model is at predicting
y_train_acc = (y_train == 1)
```

```python
y_train_rej = (y_train == 0)

# for testing purposes - later
y_test_acc = (y_test == 1)
y_test_rej = (y_test == 0)
```

In [39]:
```python
# cross validate to see performance
# lets see how well the model predicts approvals
from sklearn.model_selection import cross_val_score

cross_val_score(sgd_clf, X_train, y_train_acc, cv=3, scoring="accuracy")
```

Out[39]:
```
array([0.907758  , 0.90035543, 0.79782951])
```

about 86% accuracy. Seems like a good score, but is it true?

Confusion Matrix

In [40]:
```python
# fit a confusion matrix for use see how well it predicts 1 of the targets (approvals
from sklearn.model_selection import cross_val_predict

y_train_pred = cross_val_predict(sgd_clf, X_train, y_train_acc, cv=3)
```

In [41]:
```python
# create a confustion matrix after sgd prediction

from sklearn.metrics import confusion_matrix
confusion_matrix(y_train_acc, y_train_pred)
```

Out[41]:
```
array([[416393,  20179],
       [ 81708, 257397]], dtype=int64)
```

[TN (416,393) ,FP (20,179)]

[FN (81,708) ,TP (257,397)]

In [42]:
```python
# if the perfect prediction existed - IT would look like the below.
y_train_perfect_predictions = y_train_acc
confusion_matrix(y_train_acc, y_train_perfect_predictions)
```

Out[42]:
```
array([[436572,      0],
       [     0, 339105]], dtype=int64)
```

Precision and Recall

In [103…
```python
from sklearn.metrics import precision_score, recall_score
print("Precision score",precision_score(y_train_acc, y_train_pred),"---> What percent
print("Recall",recall_score(y_train_acc, y_train_pred), "---> What percent of the pos:
```

```
Precision score 0.9273027927486526 ---> What percent of your predictions were correct?
Recall 0.7590480824523378 ---> What percent of the positive cases did you catch?
```

combine precision and recall into a single metric called the F1 score, in particular if you need a simple way to compare two classifiers. The F1 score is the harmonic mean of precision and recall. Whereas the regular mean treats all values equally, the harmonic mean gives much more weight to low values. As a result, the classifier will only get a high F1 score if both recall and precision are high.

In [44]:
```python
from sklearn.metrics import f1_score
f1_score(y_train_acc, y_train_pred)
```

Out[44]:
```
0.8347816780474832
```

Random Forest Classifier

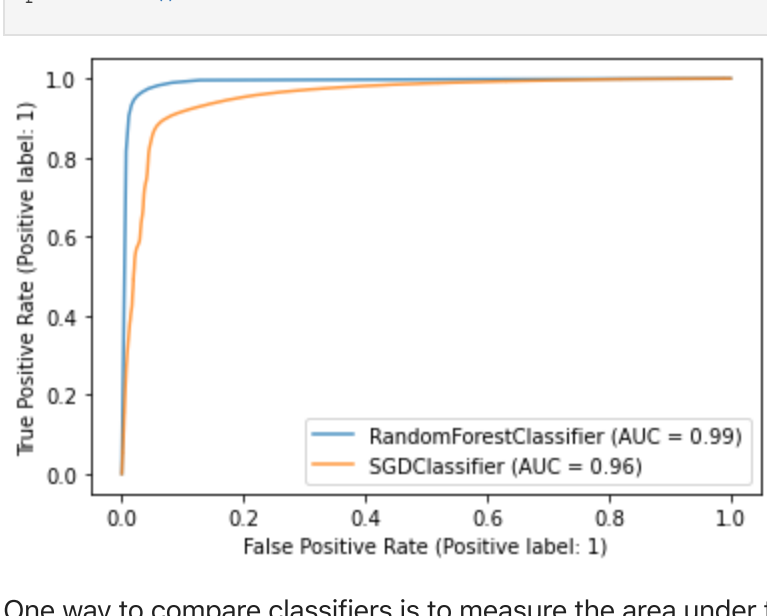
Let's train a RandomForestClassifier and compare its ROC curve and ROC AUC score to the SGDClassifier.

The receiver operating characteristic (ROC) curve is another common tool used with binary classifiers. It is very similar to the precision/recall curve, but instead of plotting precision versus recall, the ROC curve plots the true positive rate (another name for recall) against the false positive rate.

Because there are some trade-offs between precision/recall scores, ROC AUC might be the best option to evaluate Both the SGD model and the RFC model.

In [45]:
```python
# create and fit an RFC model
from sklearn.ensemble import RandomForestClassifier

forest_clf = RandomForestClassifier(random_state=42)

y_probas_forest = cross_val_predict(forest_clf, X_train, y_train_acc, cv=3, method="p
```

In [46]:
```python
# to plot a ROC curve to see score for RFC vs SGD
from sklearn.metrics import plot_roc_curve

rfc = RandomForestClassifier(n_estimators=10, random_state=42)
rfc.fit(X_train, y_train)

ax = plt.gca()
rfc_disp = plot_roc_curve(rfc, X_test, y_test, ax=ax, alpha=0.8)
sgd_clf_disp = plot_roc_curve(sgd_clf, X_test, y_test, ax=ax, alpha=0.8)
plt.show()
```



One way to compare classifiers is to measure the area under the curve (AUC). A perfect classifier will have a ROC AUC equal to 1

So here we see that RandomForest Classifier is a better model

In [47]:
```python
# Random x values for testing prediction
random_acc = X[:1]
random_rej = X[-1:]
```

In [48]:
```python
# Convert them to 2D array
random_acc = random_acc.values
random_rej = random_rej.values
```

In [104…
```python
# Predict random acc/rej with SGD model
print('Predict Acc: Should be 1--->',sgd_clf.predict(random_acc))
print('Predict Reg: Should be 0--->',sgd_clf.predict(random_rej))
```

```
Predict Acc: Should be 1---> [1]
Predict Reg: Should be 0---> [1]
```

In [105…
```python
rfc.predict(random_rej)

print('Predict Acc: Should be 1--->',rfc.predict(random_acc))
print('Predict Reg: Should be 0--->',rfc.predict(random_rej))
```

```
Predict Acc: Should be 1---> [1]
Predict Reg: Should be 0---> [0]
```

It's cleear to see that RFC performs a lot better than SGD

In [51]:
```python
# Random x values for testing prediction 100 of each target
random_acc_100 = X[:100]
random_rej_100 = X[-100:]

# Convert them to 2D array
random_acc_100 = random_acc_100.values
random_rej_100 = random_rej_100.values
```

In [69]:
```python
# Predict random acc/rej with SGD model
print('Predict SGD Acc: Should be 1 ====>',sgd_clf.predict(random_acc_100))
print('Predict SGD ReJ: Should be 0 ====>',sgd_clf.predict(random_rej_100))
print('-----------------------------------------------------')
print('-----------------------------------------------------')
print('-----------------------------------------------------')
print('Predict RFC Acc: Should be 1 ====>',rfc.predict(random_acc_100))
print('Predict RFC ReJ: Should be 0 ====>',rfc.predict(random_rej_100))
```

```
Predict SGD Acc: Should be 1 ====> [1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1
1 1 1 1 1 1 1 1 1 1 0 1
 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1]
Predict SGD ReJ: Should be 0 ====> [0 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1]
-----------------------------------------------------
-----------------------------------------------------
-----------------------------------------------------
Predict RFC Acc: Should be 1 ====> [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
Predict RFC ReJ: Should be 0 ====> [0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

Grid Search

In [67]:
```python
# Create the GridSearch estimator along with a parameter object containing the values
from sklearn.model_selection import GridSearchCV

param_grid={'max_depth': [1, 3, 5, 7, 9, 11],
            'n_estimators': [1, 21, 41, 61, 81, 101]}

grid = GridSearchCV(rfc, param_grid, verbose=3)
```

In [68]:
```python
grid.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 100 candidates, totalling 500 fits
[CV 1/5] END ...................max_depth=1, n_estimators=1; total time=   1.5s
[CV 2/5] END ...................max_depth=1, n_estimators=1; total time=   1.4s
[CV 3/5] END ...................max_depth=1, n_estimators=1; total time=   1.4s
[CV 4/5] END ...................max_depth=1, n_estimators=1; total time=   1.4s
[CV 5/5] END ...................max_depth=1, n_estimators=1; total time=   1.5s
[CV 1/5] END ..................max_depth=1, n_estimators=21; total time=   7.0s
[CV 2/5] END ..................max_depth=1, n_estimators=21; total time=   5.8s
[CV 3/5] END ..................max_depth=1, n_estimators=21; total time=   6.6s
[CV 4/5] END ..................max_depth=1, n_estimators=21; total time=   6.2s
[CV 5/5] END ..................max_depth=1, n_estimators=21; total time=   6.8s
[CV 1/5] END ..................max_depth=1, n_estimators=41; total time=  11.6s
[CV 2/5] END ..................max_depth=1, n_estimators=41; total time=  12.0s
[CV 3/5] END ..................max_depth=1, n_estimators=41; total time=  12.5s
[CV 4/5] END ..................max_depth=1, n_estimators=41; total time=  12.2s
[CV 5/5] END ..................max_depth=1, n_estimators=41; total time=  11.9s
[CV 1/5] END ..................max_depth=1, n_estimators=61; total time=  16.8s
[CV 2/5] END ..................max_depth=1, n_estimators=61; total time=  17.0s
[CV 3/5] END ..................max_depth=1, n_estimators=61; total time=  17.0s
[CV 4/5] END ..................max_depth=1, n_estimators=61; total time=  16.3s
[CV 5/5] END ..................max_depth=1, n_estimators=61; total time=  17.0s
[CV 1/5] END ..................max_depth=1, n_estimators=81; total time=  21.7s
[CV 2/5] END ..................max_depth=1, n_estimators=81; total time=  22.6s
[CV 3/5] END ..................max_depth=1, n_estimators=81; total time=  22.4s
[CV 4/5] END ..................max_depth=1, n_estimators=81; total time=  22.3s
[CV 5/5] END ..................max_depth=1, n_estimators=81; total time=  23.5s
[CV 1/5] END .................max_depth=1, n_estimators=101; total time=  27.9s
[CV 2/5] END .................max_depth=1, n_estimators=101; total time=  29.8s
[CV 3/5] END .................max_depth=1, n_estimators=101; total time=  32.4s
[CV 4/5] END .................max_depth=1, n_estimators=101; total time=  27.8s
[CV 5/5] END .................max_depth=1, n_estimators=101; total time=  28.4s
[CV 1/5] END .................max_depth=1, n_estimators=121; total time=  32.9s
[CV 2/5] END .................max_depth=1, n_estimators=121; total time=  33.5s
[CV 3/5] END .................max_depth=1, n_estimators=121; total time=  33.7s
[CV 4/5] END .................max_depth=1, n_estimators=121; total time=  34.1s
[CV 5/5] END .................max_depth=1, n_estimators=121; total time=  33.1s
[CV 1/5] END .................max_depth=1, n_estimators=141; total time=  40.1s
[CV 2/5] END .................max_depth=1, n_estimators=141; total time=  42.4s
[CV 3/5] END .................max_depth=1, n_estimators=141; total time=  38.7s
[CV 4/5] END .................max_depth=1, n_estimators=141; total time=  38.7s
[CV 5/5] END .................max_depth=1, n_estimators=141; total time=  40.1s
[CV 1/5] END .................max_depth=1, n_estimators=161; total time=  44.4s
[CV 2/5] END .................max_depth=1, n_estimators=161; total time=  45.0s
[CV 3/5] END .................max_depth=1, n_estimators=161; total time=  44.2s
[CV 4/5] END .................max_depth=1, n_estimators=161; total time=  46.9s
[CV 5/5] END .................max_depth=1, n_estimators=161; total time=  44.5s
[CV 1/5] END .................max_depth=1, n_estimators=181; total time=  50.2s
[CV 2/5] END .................max_depth=1, n_estimators=181; total time=  50.7s
[CV 3/5] END .................max_depth=1, n_estimators=181; total time=  50.7s
[CV 4/5] END .................max_depth=1, n_estimators=181; total time=  53.8s
[CV 5/5] END .................max_depth=1, n_estimators=181; total time=  51.5s
[CV 1/5] END ...................max_depth=3, n_estimators=1; total time=   1.6s
[CV 2/5] END ...................max_depth=3, n_estimators=1; total time=   1.5s
[CV 3/5] END ...................max_depth=3, n_estimators=1; total time=   1.6s
[CV 4/5] END ...................max_depth=3, n_estimators=1; total time=   1.6s
[CV 5/5] END ...................max_depth=3, n_estimators=1; total time=   1.6s
[CV 1/5] END ..................max_depth=3, n_estimators=21; total time=  12.4s
[CV 2/5] END ..................max_depth=3, n_estimators=21; total time=  10.7s
[CV 3/5] END ..................max_depth=3, n_estimators=21; total time=  11.0s
[CV 4/5] END ..................max_depth=3, n_estimators=21; total time=  10.9s
[CV 5/5] END ..................max_depth=3, n_estimators=21; total time=  12.2s
[CV 1/5] END ..................max_depth=3, n_estimators=41; total time=  20.9s
[CV 2/5] END ..................max_depth=3, n_estimators=41; total time=  20.1s
[CV 3/5] END ..................max_depth=3, n_estimators=41; total time=  19.5s
[CV 4/5] END ..................max_depth=3, n_estimators=41; total time=  19.7s
[CV 5/5] END ..................max_depth=3, n_estimators=41; total time=  19.7s
[CV 1/5] END ..................max_depth=3, n_estimators=61; total time=  28.5s
[CV 2/5] END ..................max_depth=3, n_estimators=61; total time=  29.1s
[CV 3/5] END ..................max_depth=3, n_estimators=61; total time=  29.1s
[CV 4/5] END ..................max_depth=3, n_estimators=61; total time=  29.9s
[CV 5/5] END ..................max_depth=3, n_estimators=61; total time=  30.0s
[CV 1/5] END ..................max_depth=3, n_estimators=81; total time=  39.5s
[CV 2/5] END ..................max_depth=3, n_estimators=81; total time=  39.2s
[CV 3/5] END ..................max_depth=3, n_estimators=81; total time=  38.8s
[CV 4/5] END ..................max_depth=3, n_estimators=81; total time=  40.9s
[CV 5/5] END ..................max_depth=3, n_estimators=81; total time=  39.9s
[CV 1/5] END .................max_depth=3, n_estimators=101; total time=  50.9s
[CV 2/5] END .................max_depth=3, n_estimators=101; total time=  50.4s
[CV 3/5] END .................max_depth=3, n_estimators=101; total time=  48.0s
[CV 4/5] END .................max_depth=3, n_estimators=101; total time=  50.9s
[CV 5/5] END .................max_depth=3, n_estimators=101; total time=  50.5s
[CV 1/5] END .................max_depth=3, n_estimators=121; total time=  59.9s
[CV 2/5] END .................max_depth=3, n_estimators=121; total time=  59.0s
[CV 3/5] END .................max_depth=3, n_estimators=121; total time=  58.3s
[CV 4/5] END .................max_depth=3, n_estimators=121; total time= 1.1min
[CV 5/5] END .................max_depth=3, n_estimators=121; total time=  58.9s
[CV 1/5] END .................max_depth=3, n_estimators=141; total time= 1.1min
[CV 2/5] END .................max_depth=3, n_estimators=141; total time= 1.1min
[CV 3/5] END .................max_depth=3, n_estimators=141; total time= 1.2min
[CV 4/5] END .................max_depth=3, n_estimators=141; total time= 1.2min
[CV 5/5] END .................max_depth=3, n_estimators=141; total time= 1.2min
[CV 1/5] END .................max_depth=3, n_estimators=161; total time= 1.4min
[CV 2/5] END .................max_depth=3, n_estimators=161; total time= 1.4min
[CV 3/5] END .................max_depth=3, n_estimators=161; total time= 1.3min
[CV 4/5] END .................max_depth=3, n_estimators=161; total time= 1.4min
[CV 5/5] END .................max_depth=3, n_estimators=161; total time= 1.4min
[CV 1/5] END .................max_depth=3, n_estimators=181; total time= 1.5min
[CV 2/5] END .................max_depth=3, n_estimators=181; total time= 1.5min
[CV 3/5] END .................max_depth=3, n_estimators=181; total time= 1.5min
[CV 4/5] END .................max_depth=3, n_estimators=181; total time= 1.5min
[CV 5/5] END .................max_depth=3, n_estimators=181; total time= 1.5min
[CV 1/5] END ...................max_depth=5, n_estimators=1; total time=   1.9s
[CV 2/5] END ...................max_depth=5, n_estimators=1; total time=   1.9s
[CV 3/5] END ...................max_depth=5, n_estimators=1; total time=   2.0s
[CV 4/5] END ...................max_depth=5, n_estimators=1; total time=   1.7s
[CV 5/5] END ...................max_depth=5, n_estimators=1; total time=   1.3s
[CV 1/5] END ..................max_depth=5, n_estimators=21; total time=  15.7s
[CV 2/5] END ..................max_depth=5, n_estimators=21; total time=  15.9s
[CV 3/5] END ..................max_depth=5, n_estimators=21; total time=  15.9s
[CV 4/5] END ..................max_depth=5, n_estimators=21; total time=  16.1s
[CV 5/5] END ..................max_depth=5, n_estimators=21; total time=  16.3s
[CV 1/5] END ..................max_depth=5, n_estimators=41; total time=  29.6s
[CV 2/5] END ..................max_depth=5, n_estimators=41; total time=  29.4s
[CV 3/5] END ..................max_depth=5, n_estimators=41; total time=  30.5s
[CV 4/5] END ..................max_depth=5, n_estimators=41; total time=  33.3s
[CV 5/5] END ..................max_depth=5, n_estimators=41; total time=  18.3s
[CV 1/5] END ..................max_depth=5, n_estimators=61; total time=  26.1s
[CV 2/5] END ..................max_depth=5, n_estimators=61; total time=  22.7s
[CV 3/5] END ..................max_depth=5, n_estimators=61; total time=  23.0s
[CV 4/5] END ..................max_depth=5, n_estimators=61; total time=  22.5s
[CV 5/5] END ..................max_depth=5, n_estimators=61; total time=  22.7s
[CV 1/5] END ..................max_depth=5, n_estimators=81; total time=  29.7s
[CV 2/5] END ..................max_depth=5, n_estimators=81; total time=  30.2s
[CV 3/5] END ..................max_depth=5, n_estimators=81; total time=  29.9s
[CV 4/5] END ..................max_depth=5, n_estimators=81; total time=  30.0s
[CV 5/5] END ..................max_depth=5, n_estimators=81; total time=  30.3s
[CV 1/5] END .................max_depth=5, n_estimators=101; total time=  37.2s
[CV 2/5] END .................max_depth=5, n_estimators=101; total time=  38.4s
[CV 3/5] END .................max_depth=5, n_estimators=101; total time=  37.4s
[CV 4/5] END .................max_depth=5, n_estimators=101; total time=  37.2s
[CV 5/5] END .................max_depth=5, n_estimators=101; total time=  37.0s
[CV 1/5] END .................max_depth=5, n_estimators=121; total time=  43.6s
[CV 2/5] END .................max_depth=5, n_estimators=121; total time=  44.4s
[CV 3/5] END .................max_depth=5, n_estimators=121; total time=  43.6s
[CV 4/5] END .................max_depth=5, n_estimators=121; total time=  45.1s
[CV 5/5] END .................max_depth=5, n_estimators=121; total time=  52.6s
[CV 1/5] END .................max_depth=5, n_estimators=141; total time=  58.9s
[CV 2/5] END .................max_depth=5, n_estimators=141; total time= 1.0min
[CV 3/5] END .................max_depth=5, n_estimators=141; total time=  59.4s
[CV 4/5] END .................max_depth=5, n_estimators=141; total time= 1.0min
[CV 5/5] END .................max_depth=5, n_estimators=141; total time= 1.0min
[CV 1/5] END .................max_depth=5, n_estimators=161; total time= 1.1min
[CV 2/5] END .................max_depth=5, n_estimators=161; total time= 1.1min
[CV 3/5] END .................max_depth=5, n_estimators=161; total time= 1.1min
[CV 4/5] END .................max_depth=5, n_estimators=161; total time= 1.0min
[CV 5/5] END .................max_depth=5, n_estimators=161; total time=  56.2s
[CV 1/5] END .................max_depth=5, n_estimators=181; total time= 1.0min
[CV 2/5] END .................max_depth=5, n_estimators=181; total time= 1.0min
[CV 3/5] END .................max_depth=5, n_estimators=181; total time= 1.0min
[CV 4/5] END .................max_depth=5, n_estimators=181; total time= 1.1min
[CV 5/5] END .................max_depth=5, n_estimators=181; total time= 1.0min
[CV 1/5] END ...................max_depth=7, n_estimators=1; total time=   0.9s
[CV 2/5] END ...................max_depth=7, n_estimators=1; total time=   1.0s
[CV 3/5] END ...................max_depth=7, n_estimators=1; total time=   1.1s
[CV 4/5] END ...................max_depth=7, n_estimators=1; total time=   0.9s
[CV 5/5] END ...................max_depth=7, n_estimators=1; total time=   0.9s
[CV 1/5] END ..................max_depth=7, n_estimators=21; total time=   9.8s
[CV 2/5] END ..................max_depth=7, n_estimators=21; total time=   9.8s
[CV 3/5] END ..................max_depth=7, n_estimators=21; total time=   9.8s
[CV 4/5] END ..................max_depth=7, n_estimators=21; total time=   9.8s
[CV 5/5] END ..................max_depth=7, n_estimators=21; total time=   9.6s
[CV 1/5] END ..................max_depth=7, n_estimators=41; total time=  18.4s
[CV 2/5] END ..................max_depth=7, n_estimators=41; total time=  19.0s
[CV 3/5] END ..................max_depth=7, n_estimators=41; total time=  18.5s
[CV 4/5] END ..................max_depth=7, n_estimators=41; total time=  19.0s
[CV 5/5] END ..................max_depth=7, n_estimators=41; total time=  18.4s
[CV 1/5] END ..................max_depth=7, n_estimators=61; total time=  28.2s
[CV 2/5] END ..................max_depth=7, n_estimators=61; total time=  28.0s
[CV 3/5] END ..................max_depth=7, n_estimators=61; total time=  27.6s
[CV 4/5] END ..................max_depth=7, n_estimators=61; total time=  29.1s
[CV 5/5] END ..................max_depth=7, n_estimators=61; total time=  28.4s
[CV 1/5] END ..................max_depth=7, n_estimators=81; total time=  36.8s
[CV 2/5] END ..................max_depth=7, n_estimators=81; total time=  37.4s
[CV 3/5] END ..................max_depth=7, n_estimators=81; total time=  37.1s
[CV 4/5] END ..................max_depth=7, n_estimators=81; total time=  36.7s
[CV 5/5] END ..................max_depth=7, n_estimators=81; total time=  37.3s
[CV 1/5] END .................max_depth=7, n_estimators=101; total time=  45.7s
[CV 2/5] END .................max_depth=7, n_estimators=101; total time=  46.0s
[CV 3/5] END .................max_depth=7, n_estimators=101; total time=  46.4s
[CV 4/5] END .................max_depth=7, n_estimators=101; total time=  45.4s
[CV 5/5] END .................max_depth=7, n_estimators=101; total time=  45.5s
[CV 1/5] END .................max_depth=7, n_estimators=121; total time=  54.3s
[CV 2/5] END .................max_depth=7, n_estimators=121; total time=  54.7s
[CV 3/5] END .................max_depth=7, n_estimators=121; total time=  54.3s
[CV 4/5] END .................max_depth=7, n_estimators=121; total time=  53.4s
[CV 5/5] END .................max_depth=7, n_estimators=121; total time=  54.5s
[CV 1/5] END .................max_depth=7, n_estimators=141; total time= 1.0min
[CV 2/5] END .................max_depth=7, n_estimators=141; total time= 1.1min
[CV 3/5] END .................max_depth=7, n_estimators=141; total time= 1.1min
[CV 4/5] END .................max_depth=7, n_estimators=141; total time= 1.0min
[CV 5/5] END .................max_depth=7, n_estimators=141; total time= 1.0min
[CV 1/5] END .................max_depth=7, n_estimators=161; total time= 1.2min
[CV 2/5] END .................max_depth=7, n_estimators=161; total time= 1.2min
[CV 3/5] END .................max_depth=7, n_estimators=161; total time= 1.2min
[CV 4/5] END .................max_depth=7, n_estimators=161; total time= 1.2min
[CV 5/5] END .................max_depth=7, n_estimators=161; total time= 1.2min
[CV 1/5] END .................max_depth=7, n_estimators=181; total time= 1.3min
[CV 2/5] END .................max_depth=7, n_estimators=181; total time= 1.4min
[CV 3/5] END .................max_depth=7, n_estimators=181; total time= 1.3min
[CV 4/5] END .................max_depth=7, n_estimators=181; total time= 1.4min
[CV 5/5] END .................max_depth=7, n_estimators=181; total time= 1.3min
[CV 1/5] END ...................max_depth=9, n_estimators=1; total time=   1.2s
[CV 2/5] END ...................max_depth=9, n_estimators=1; total time=   1.4s
[CV 3/5] END ...................max_depth=9, n_estimators=1; total time=   1.3s
[CV 4/5] END ...................max_depth=9, n_estimators=1; total time=   1.1s
[CV 5/5] END ...................max_depth=9, n_estimators=1; total time=   1.0s
[CV 1/5] END ..................max_depth=9, n_estimators=21; total time=  11.7s
[CV 2/5] END ..................max_depth=9, n_estimators=21; total time=  11.8s
[CV 3/5] END ..................max_depth=9, n_estimators=21; total time=  12.0s
[CV 4/5] END ..................max_depth=9, n_estimators=21; total time=  11.6s
[CV 5/5] END ..................max_depth=9, n_estimators=21; total time=  11.8s
[CV 1/5] END ..................max_depth=9, n_estimators=41; total time=  22.5s
[CV 2/5] END ..................max_depth=9, n_estimators=41; total time=  22.5s
[CV 3/5] END ..................max_depth=9, n_estimators=41; total time=  23.2s
[CV 4/5] END ..................max_depth=9, n_estimators=41; total time=  22.6s
[CV 5/5] END ..................max_depth=9, n_estimators=41; total time=  23.4s
[CV 1/5] END ..................max_depth=9, n_estimators=61; total time=  34.2s
[CV 2/5] END ..................max_depth=9, n_estimators=61; total time=  35.3s
[CV 3/5] END ..................max_depth=9, n_estimators=61; total time=  35.4s
[CV 4/5] END ..................max_depth=9, n_estimators=61; total time=  34.0s
[CV 5/5] END ..................max_depth=9, n_estimators=61; total time=  34.6s
[CV 1/5] END ..................max_depth=9, n_estimators=81; total time=  44.9s
[CV 2/5] END ..................max_depth=9, n_estimators=81; total time=  45.0s
[CV 3/5] END ..................max_depth=9, n_estimators=81; total time=  46.0s
[CV 4/5] END ..................max_depth=9, n_estimators=81; total time=  45.3s
[CV 5/5] END ..................max_depth=9, n_estimators=81; total time=  45.4s
[CV 1/5] END .................max_depth=9, n_estimators=101; total time=  56.1s
[CV 2/5] END .................max_depth=9, n_estimators=101; total time=  55.8s
[CV 3/5] END .................max_depth=9, n_estimators=101; total time=  56.5s
[CV 4/5] END .................max_depth=9, n_estimators=101; total time=  59.6s
[CV 5/5] END .................max_depth=9, n_estimators=101; total time=  56.3s
[CV 1/5] END .................max_depth=9, n_estimators=121; total time= 1.1min
[CV 2/5] END .................max_depth=9, n_estimators=121; total time= 1.1min
[CV 3/5] END .................max_depth=9, n_estimators=121; total time= 1.1min
[CV 4/5] END .................max_depth=9, n_estimators=121; total time= 1.1min
[CV 5/5] END .................max_depth=9, n_estimators=121; total time= 1.1min
[CV 1/5] END .................max_depth=9, n_estimators=141; total time= 1.3min
[CV 2/5] END .................max_depth=9, n_estimators=141; total time= 1.3min
[CV 3/5] END .................max_depth=9, n_estimators=141; total time= 1.3min
[CV 4/5] END .................max_depth=9, n_estimators=141; total time= 1.3min
[CV 5/5] END .................max_depth=9, n_estimators=141; total time= 1.3min
[CV 1/5] END .................max_depth=9, n_estimators=161; total time= 1.5min
[CV 2/5] END .................max_depth=9, n_estimators=161; total time= 1.5min
[CV 3/5] END .................max_depth=9, n_estimators=161; total time= 1.5min
[CV 4/5] END .................max_depth=9, n_estimators=161; total time= 1.5min
[CV 5/5] END .................max_depth=9, n_estimators=161; total time= 1.5min
[CV 1/5] END .................max_depth=9, n_estimators=181; total time= 1.6min
[CV 2/5] END .................max_depth=9, n_estimators=181; total time= 1.8min
[CV 3/5] END .................max_depth=9, n_estimators=181; total time= 1.7min
[CV 4/5] END .................max_depth=9, n_estimators=181; total time= 1.7min
[CV 5/5] END .................max_depth=9, n_estimators=181; total time= 1.7min
[CV 1/5] END ..................max_depth=11, n_estimators=1; total time=   1.2s
[CV 2/5] END ..................max_depth=11, n_estimators=1; total time=   1.1s
[CV 3/5] END ..................max_depth=11, n_estimators=1; total time=   1.1s
[CV 4/5] END ..................max_depth=11, n_estimators=1; total time=   1.2s
[CV 5/5] END ..................max_depth=11, n_estimators=1; total time=   1.1s
[CV 1/5] END .................max_depth=11, n_estimators=21; total time=  14.0s
[CV 2/5] END .................max_depth=11, n_estimators=21; total time=  13.8s
[CV 3/5] END .................max_depth=11, n_estimators=21; total time=  13.9s
[CV 4/5] END .................max_depth=11, n_estimators=21; total time=  13.5s
[CV 5/5] END .................max_depth=11, n_estimators=21; total time=  14.1s
[CV 1/5] END .................max_depth=11, n_estimators=41; total time=  26.7s
[CV 2/5] END .................max_depth=11, n_estimators=41; total time=  26.6s
[CV 3/5] END .................max_depth=11, n_estimators=41; total time=  27.2s
[CV 4/5] END .................max_depth=11, n_estimators=41; total time=  26.5s
[CV 5/5] END .................max_depth=11, n_estimators=41; total time=  27.0s
[CV 1/5] END .................max_depth=11, n_estimators=61; total time=  40.0s
[CV 2/5] END .................max_depth=11, n_estimators=61; total time=  39.4s
[CV 3/5] END .................max_depth=11, n_estimators=61; total time=  40.7s
[CV 4/5] END .................max_depth=11, n_estimators=61; total time=  40.3s
[CV 5/5] END .................max_depth=11, n_estimators=61; total time=  40.3s
[CV 1/5] END .................max_depth=11, n_estimators=81; total time=  53.0s
[CV 2/5] END .................max_depth=11, n_estimators=81; total time=  52.7s
[CV 3/5] END .................max_depth=11, n_estimators=81; total time=  52.9s
[CV 4/5] END .................max_depth=11, n_estimators=81; total time=  52.9s
[CV 5/5] END .................max_depth=11, n_estimators=81; total time=  52.7s
[CV 1/5] END ................max_depth=11, n_estimators=101; total time= 1.1min
[CV 2/5] END ................max_depth=11, n_estimators=101; total time= 1.1min
[CV 3/5] END ................max_depth=11, n_estimators=101; total time= 1.1min
[CV 4/5] END ................max_depth=11, n_estimators=101; total time= 1.1min
[CV 5/5] END ................max_depth=11, n_estimators=101; total time= 1.1min
[CV 1/5] END ................max_depth=11, n_estimators=121; total time= 1.3min
[CV 2/5] END ................max_depth=11, n_estimators=121; total time= 1.3min
[CV 3/5] END ................max_depth=11, n_estimators=121; total time= 1.3min
[CV 4/5] END ................max_depth=11, n_estimators=121; total time= 1.3min
[CV 5/5] END ................max_depth=11, n_estimators=121; total time= 1.3min
[CV 1/5] END ................max_depth=11, n_estimators=141; total time= 1.5min
[CV 2/5] END ................max_depth=11, n_estimators=141; total time= 1.5min
[CV 3/5] END ................max_depth=11, n_estimators=141; total time= 1.5min
[CV 4/5] END ................max_depth=11, n_estimators=141; total time= 1.5min
[CV 5/5] END ................max_depth=11, n_estimators=141; total time= 1.5min
[CV 1/5] END ................max_depth=11, n_estimators=161; total time= 1.7min
[CV 2/5] END ................max_depth=11, n_estimators=161; total time= 1.7min
[CV 3/5] END ................max_depth=11, n_estimators=161; total time= 1.7min
[CV 4/5] END ................max_depth=11, n_estimators=161; total time= 1.7min
[CV 5/5] END ................max_depth=11, n_estimators=161; total time= 1.7min
[CV 1/5] END ................max_depth=11, n_estimators=181; total time= 1.9min
[CV 2/5] END ................max_depth=11, n_estimators=181; total time= 1.9min
[CV 3/5] END ................max_depth=11, n_estimators=181; total time= 1.9min
[CV 4/5] END ................max_depth=11, n_estimators=181; total time= 1.9min
[CV 5/5] END ................max_depth=11, n_estimators=181; total time= 1.9min
[CV 1/5] END ..................max_depth=13, n_estimators=1; total time=   1.2s
[CV 2/5] END ..................max_depth=13, n_estimators=1; total time=   1.2s
[CV 3/5] END ..................max_depth=13, n_estimators=1; total time=   1.1s
[CV 4/5] END ..................max_depth=13, n_estimators=1; total time=   1.2s
[CV 5/5] END ..................max_depth=13, n_estimators=1; total time=   1.2s
[CV 1/5] END .................max_depth=13, n_estimators=21; total time=  15.3s
[CV 2/5] END .................max_depth=13, n_estimators=21; total time=  15.9s
[CV 3/5] END .................max_depth=13, n_estimators=21; total time=  15.7s
[CV 4/5] END .................max_depth=13, n_estimators=21; total time=  15.1s
[CV 5/5] END .................max_depth=13, n_estimators=21; total time=  15.9s
[CV 1/5] END .................max_depth=13, n_estimators=41; total time=  30.2s
[CV 2/5] END .................max_depth=13, n_estimators=41; total time=  30.5s
[CV 3/5] END .................max_depth=13, n_estimators=41; total time=  29.9s
[CV 4/5] END .................max_depth=13, n_estimators=41; total time=  29.6s
[CV 5/5] END .................max_depth=13, n_estimators=41; total time=  30.2s
[CV 1/5] END .................max_depth=13, n_estimators=61; total time=  44.8s
[CV 2/5] END .................max_depth=13, n_estimators=61; total time=  45.4s
[CV 3/5] END .................max_depth=13, n_estimators=61; total time=  45.4s
[CV 4/5] END .................max_depth=13, n_estimators=61; total time=  45.6s
[CV 5/5] END .................max_depth=13, n_estimators=61; total time=  44.6s
[CV 1/5] END .................max_depth=13, n_estimators=81; total time=  59.1s
[CV 2/5] END .................max_depth=13, n_estimators=81; total time= 1.0min
[CV 3/5] END .................max_depth=13, n_estimators=81; total time=  59.9s
[CV 4/5] END .................max_depth=13, n_estimators=81; total time=  59.3s
[CV 5/5] END .................max_depth=13, n_estimators=81; total time=  59.0s
[CV 1/5] END ................max_depth=13, n_estimators=101; total time= 1.2min
[CV 2/5] END ................max_depth=13, n_estimators=101; total time= 1.2min
[CV 3/5] END ................max_depth=13, n_estimators=101; total time= 1.2min
[CV 4/5] END ................max_depth=13, n_estimators=101; total time= 1.2min
[CV 5/5] END ................max_depth=13, n_estimators=101; total time= 1.2min
[CV 1/5] END ................max_depth=13, n_estimators=121; total time= 1.4min
[CV 2/5] END ................max_depth=13, n_estimators=121; total time= 1.5min
[CV 3/5] END ................max_depth=13, n_estimators=121; total time= 1.5min
[CV 4/5] END ................max_depth=13, n_estimators=121; total time= 1.4min
[CV 5/5] END ................max_depth=13, n_estimators=121; total time= 1.4min
[CV 1/5] END ................max_depth=13, n_estimators=141; total time= 1.7min
[CV 2/5] END ................max_depth=13, n_estimators=141; total time= 1.7min
[CV 3/5] END ................max_depth=13, n_estimators=141; total time= 1.7min
[CV 4/5] END ................max_depth=13, n_estimators=141; total time= 1.7min
[CV 5/5] END ................max_depth=13, n_estimators=141; total time= 1.7min
[CV 1/5] END ................max_depth=13, n_estimators=161; total time= 1.9min
[CV 2/5] END ................max_depth=13, n_estimators=161; total time= 1.9min
[CV 3/5] END ................max_depth=13, n_estimators=161; total time= 2.2min
[CV 4/5] END ................max_depth=13, n_estimators=161; total time= 2.1min
[CV 5/5] END ................max_depth=13, n_estimators=161; total time= 1.9min
[CV 1/5] END ................max_depth=13, n_estimators=181; total time= 2.2min
[CV 2/5] END ................max_depth=13, n_estimators=181; total time= 2.2min
[CV 3/5] END ................max_depth=13, n_estimators=181; total time= 2.2min
[CV 4/5] END ................max_depth=13, n_estimators=181; total time= 2.2min
[CV 5/5] END ................max_depth=13, n_estimators=181; total time= 2.1min
[CV 1/5] END ..................max_depth=15, n_estimators=1; total time=   1.2s
[CV 2/5] END ..................max_depth=15, n_estimators=1; total time=   1.4s
[CV 3/5] END ..................max_depth=15, n_estimators=1; total time=   1.3s
[CV 4/5] END ..................max_depth=15, n_estimators=1; total time=   1.3s
[CV 5/5] END ..................max_depth=15, n_estimators=1; total time=   1.2s
[CV 1/5] END .................max_depth=15, n_estimators=21; total time=  17.3s
[CV 2/5] END .................max_depth=15, n_estimators=21; total time=  17.6s
[CV 3/5] END .................max_depth=15, n_estimators=21; total time=  18.3s
[CV 4/5] END .................max_depth=15, n_estimators=21; total time=  17.0s
[CV 5/5] END .................max_depth=15, n_estimators=21; total time=  16.9s
[CV 1/5] END .................max_depth=15, n_estimators=41; total time=  33.5s
[CV 2/5] END .................max_depth=15, n_estimators=41; total time=  33.4s
[CV 3/5] END .................max_depth=15, n_estimators=41; total time=  32.9s
[CV 4/5] END .................max_depth=15, n_estimators=41; total time=  34.0s
[CV 5/5] END .................max_depth=15, n_estimators=41; total time=  32.7s
[CV 1/5] END .................max_depth=15, n_estimators=61; total time=  49.9s
[CV 2/5] END .................max_depth=15, n_estimators=61; total time=  50.5s
[CV 3/5] END .................max_depth=15, n_estimators=61; total time=  49.5s
[CV 4/5] END .................max_depth=15, n_estimators=61; total time=  50.2s
[CV 5/5] END .................max_depth=15, n_estimators=61; total time=  50.2s
[CV 1/5] END .................max_depth=15, n_estimators=81; total time= 1.1min
[CV 2/5] END .................max_depth=15, n_estimators=81; total time= 1.1min
[CV 3/5] END .................max_depth=15, n_estimators=81; total time= 1.1min
[CV 4/5] END .................max_depth=15, n_estimators=81; total time= 1.1min
[CV 5/5] END .................max_depth=15, n_estimators=81; total time= 1.1min
[CV 1/5] END ................max_depth=15, n_estimators=101; total time= 1.4min
[CV 2/5] END ................max_depth=15, n_estimators=101; total time= 1.4min
[CV 3/5] END ................max_depth=15, n_estimators=101; total time= 1.3min
[CV 4/5] END ................max_depth=15, n_estimators=101; total time= 1.4min
[CV 5/5] END ................max_depth=15, n_estimators=101; total time= 1.4min
[CV 1/5] END ................max_depth=15, n_estimators=121; total time= 1.6min
[CV 2/5] END ................max_depth=15, n_estimators=121; total time= 1.6min
[CV 3/5] END ................max_depth=15, n_estimators=121; total time= 1.6min
[CV 4/5] END ................max_depth=15, n_estimators=121; total time= 1.6min
[CV 5/5] END ................max_depth=15, n_estimators=121; total time= 1.6min
[CV 1/5] END ................max_depth=15, n_estimators=141; total time= 1.9min
[CV 2/5] END ................max_depth=15, n_estimators=141; total time= 1.9min
[CV 3/5] END ................max_depth=15, n_estimators=141; total time= 1.9min
[CV 4/5] END ................max_depth=15, n_estimators=141; total time= 1.9min
[CV 5/5] END ................max_depth=15, n_estimators=141; total time= 1.9min
[CV 1/5] END ................max_depth=15, n_estimators=161; total time= 2.1min
[CV 2/5] END ................max_depth=15, n_estimators=161; total time= 2.1min
[CV 3/5] END ................max_depth=15, n_estimators=161; total time= 2.1min
[CV 4/5] END ................max_depth=15, n_estimators=161; total time= 2.1min
[CV 5/5] END ................max_depth=15, n_estimators=161; total time= 2.1min
[CV 1/5] END ................max_depth=15, n_estimators=181; total time= 2.4min
[CV 2/5] END ................max_depth=15, n_estimators=181; total time= 2.4min
[CV 3/5] END ................max_depth=15, n_estimators=181; total time= 2.4min
[CV 4/5] END ................max_depth=15, n_estimators=181; total time= 2.4min
[CV 5/5] END ................max_depth=15, n_estimators=181; total time= 2.4min
[CV 1/5] END ..................max_depth=17, n_estimators=1; total time=   1.5s
[CV 2/5] END ..................max_depth=17, n_estimators=1; total time=   1.6s
[CV 3/5] END ..................max_depth=17, n_estimators=1; total time=   1.4s
[CV 4/5] END ..................max_depth=17, n_estimators=1; total time=   1.4s
[CV 5/5] END ..................max_depth=17, n_estimators=1; total time=   1.4s
[CV 1/5] END .................max_depth=17, n_estimators=21; total time=  18.8s
[CV 2/5] END .................max_depth=17, n_estimators=21; total time=  18.8s
[CV 3/5] END .................max_depth=17, n_estimators=21; total time=  19.2s
[CV 4/5] END .................max_depth=17, n_estimators=21; total time=  19.2s
[CV 5/5] END .................max_depth=17, n_estimators=21; total time=  19.4s
[CV 1/5] END .................max_depth=17, n_estimators=41; total time=  36.4s
[CV 2/5] END .................max_depth=17, n_estimators=41; total time=  37.4s
[CV 3/5] END .................max_depth=17, n_estimators=41; total time=  35.9s
[CV 4/5] END .................max_depth=17, n_estimators=41; total time=  36.9s
[CV 5/5] END .................max_depth=17, n_estimators=41; total time=  36.5s
[CV 1/5] END .................max_depth=17, n_estimators=61; total time=  53.9s
[CV 2/5] END .................max_depth=17, n_estimators=61; total time=  55.3s
[CV 3/5] END .................max_depth=17, n_estimators=61; total time=  54.6s
[CV 4/5] END .................max_depth=17, n_estimators=61; total time=  55.2s
[CV 5/5] END .................max_depth=17, n_estimators=61; total time=  55.1s
[CV 1/5] END .................max_depth=17, n_estimators=81; total time= 1.2min
[CV 2/5] END .................max_depth=17, n_estimators=81; total time= 1.2min
[CV 3/5] END .................max_depth=17, n_estimators=81; total time= 1.2min
[CV 4/5] END .................max_depth=17, n_estimators=81; total time= 1.2min
[CV 5/5] END .................max_depth=17, n_estimators=81; total time= 1.2min
[CV 1/5] END ................max_depth=17, n_estimators=101; total time= 1.5min
[CV 2/5] END ................max_depth=17, n_estimators=101; total time= 1.5min
[CV 3/5] END ................max_depth=17, n_estimators=101; total time= 1.5min
[CV 4/5] END ................max_depth=17, n_estimators=101; total time= 1.5min
[CV 5/5] END ................max_depth=17, n_estimators=101; total time= 1.5min
[CV 1/5] END ................max_depth=17, n_estimators=121; total time= 1.7min
[CV 2/5] END ................max_depth=17, n_estimators=121; total time= 1.8min
[CV 3/5] END ................max_depth=17, n_estimators=121; total time= 1.8min
[CV 4/5] END ................max_depth=17, n_estimators=121; total time= 1.7min
[CV 5/5] END ................max_depth=17, n_estimators=121; total time= 1.8min
[CV 1/5] END ................max_depth=17, n_estimators=141; total time= 2.0min
[CV 2/5] END ................max_depth=17, n_estimators=141; total time= 2.1min
[CV 3/5] END ................max_depth=17, n_estimators=141; total time= 2.1min
[CV 4/5] END ................max_depth=17, n_estimators=141; total time= 2.0min
[CV 5/5] END ................max_depth=17, n_estimators=141; total time= 2.0min
[CV 1/5] END ................max_depth=17, n_estimators=161; total time= 2.3min
[CV 2/5] END ................max_depth=17, n_estimators=161; total time= 2.4min
[CV 3/5] END ................max_depth=17, n_estimators=161; total time= 2.4min
[CV 4/5] END ................max_depth=17, n_estimators=161; total time= 2.3min
[CV 5/5] END ................max_depth=17, n_estimators=161; total time= 2.3min
[CV 1/5] END ................max_depth=17, n_estimators=181; total time= 2.6min
[CV 2/5] END ................max_depth=17, n_estimators=181; total time= 2.6min
[CV 3/5] END ................max_depth=17, n_estimators=181; total time= 2.6min
[CV 4/5] END ................max_depth=17, n_estimators=181; total time= 2.6min
[CV 5/5] END ................max_depth=17, n_estimators=181; total time= 2.6min
[CV 1/5] END ..................max_depth=19, n_estimators=1; total time=   1.5s
[CV 2/5] END ..................max_depth=19, n_estimators=1; total time=   1.4s
[CV 3/5] END ..................max_depth=19, n_estimators=1; total time=   1.4s
[CV 4/5] END ..................max_depth=19, n_estimators=1; total time=   1.5s
[CV 5/5] END ..................max_depth=19, n_estimators=1; total time=   1.6s
[CV 1/5] END .................max_depth=19, n_estimators=21; total time=  19.9s
[CV 2/5] END .................max_depth=19, n_estimators=21; total time=  19.9s
[CV 3/5] END .................max_depth=19, n_estimators=21; total time=  20.2s
[CV 4/5] END .................max_depth=19, n_estimators=21; total time=  20.8s
[CV 5/5] END .................max_depth=19, n_estimators=21; total time=  24.6s
[CV 1/5] END .................max_depth=19, n_estimators=41; total time=  40.2s
[CV 2/5] END .................max_depth=19, n_estimators=41; total time=  38.8s
[CV 3/5] END .................max_depth=19, n_estimators=41; total time=  39.4s
[CV 4/5] END .................max_depth=19, n_estimators=41; total time=  39.2s
[CV 5/5] END .................max_depth=19, n_estimators=41; total time=  38.4s
[CV 1/5] END .................max_depth=19, n_estimators=61; total time=  58.9s
[CV 2/5] END .................max_depth=19, n_estimators=61; total time=  58.0s
[CV 3/5] END .................max_depth=19, n_estimators=61; total time=  58.5s
[CV 4/5] END .................max_depth=19, n_estimators=61; total time=  58.4s
[CV 5/5] END .................max_depth=19, n_estimators=61; total time=  58.3s
[CV 1/5] END .................max_depth=19, n_estimators=81; total time= 1.3min
[CV 2/5] END .................max_depth=19, n_estimators=81; total time= 1.3min
[CV 3/5] END .................max_depth=19, n_estimators=81; total time= 1.3min
[CV 4/5] END .................max_depth=19, n_estimators=81; total time= 1.3min
[CV 5/5] END .................max_depth=19, n_estimators=81; total time= 1.3min
[CV 1/5] END ................max_depth=19, n_estimators=101; total time= 1.6min
[CV 2/5] END ................max_depth=19, n_estimators=101; total time= 1.6min
[CV 3/5] END ................max_depth=19, n_estimators=101; total time= 1.6min
[CV 4/5] END ................max_depth=19, n_estimators=101; total time= 1.6min
[CV 5/5] END ................max_depth=19, n_estimators=101; total time= 1.6min
[CV 1/5] END ................max_depth=19, n_estimators=121; total time= 1.9min
[CV 2/5] END ................max_depth=19, n_estimators=121; total time= 1.9min
[CV 3/5] END ................max_depth=19, n_estimators=121; total time= 1.9min
[CV 4/5] END ................max_depth=19, n_estimators=121; total time= 1.9min
[CV 5/5] END ................max_depth=19, n_estimators=121; total time= 1.9min
[CV 1/5] END ................max_depth=19, n_estimators=141; total time= 2.2min
[CV 2/5] END ................max_depth=19, n_estimators=141; total time= 2.2min
[CV 3/5] END ................max_depth=19, n_estimators=141; total time= 2.2min
[CV 4/5] END ................max_depth=19, n_estimators=141; total time= 2.2min
[CV 5/5] END ................max_depth=19, n_estimators=141; total time= 2.2min
[CV 1/5] END ................max_depth=19, n_estimators=161; total time= 2.5min
[CV 2/5] END ................max_depth=19, n_estimators=161; total time= 2.5min
[CV 3/5] END ................max_depth=19, n_estimators=161; total time= 2.5min
[CV 4/5] END ................max_depth=19, n_estimators=161; total time= 2.5min
[CV 5/5] END ................max_depth=19, n_estimators=161; total time= 2.5min
[CV 1/5] END ................max_depth=19, n_estimators=181; total time= 2.8min
[CV 2/5] END ................max_depth=19, n_estimators=181; total time= 2.8min
[CV 3/5] END ................max_depth=19, n_estimators=181; total time= 2.8min
[CV 4/5] END ................max_depth=19, n_estimators=181; total time= 2.8min
[CV 5/5] END ................max_depth=19, n_estimators=181; total time= 2.8min
```

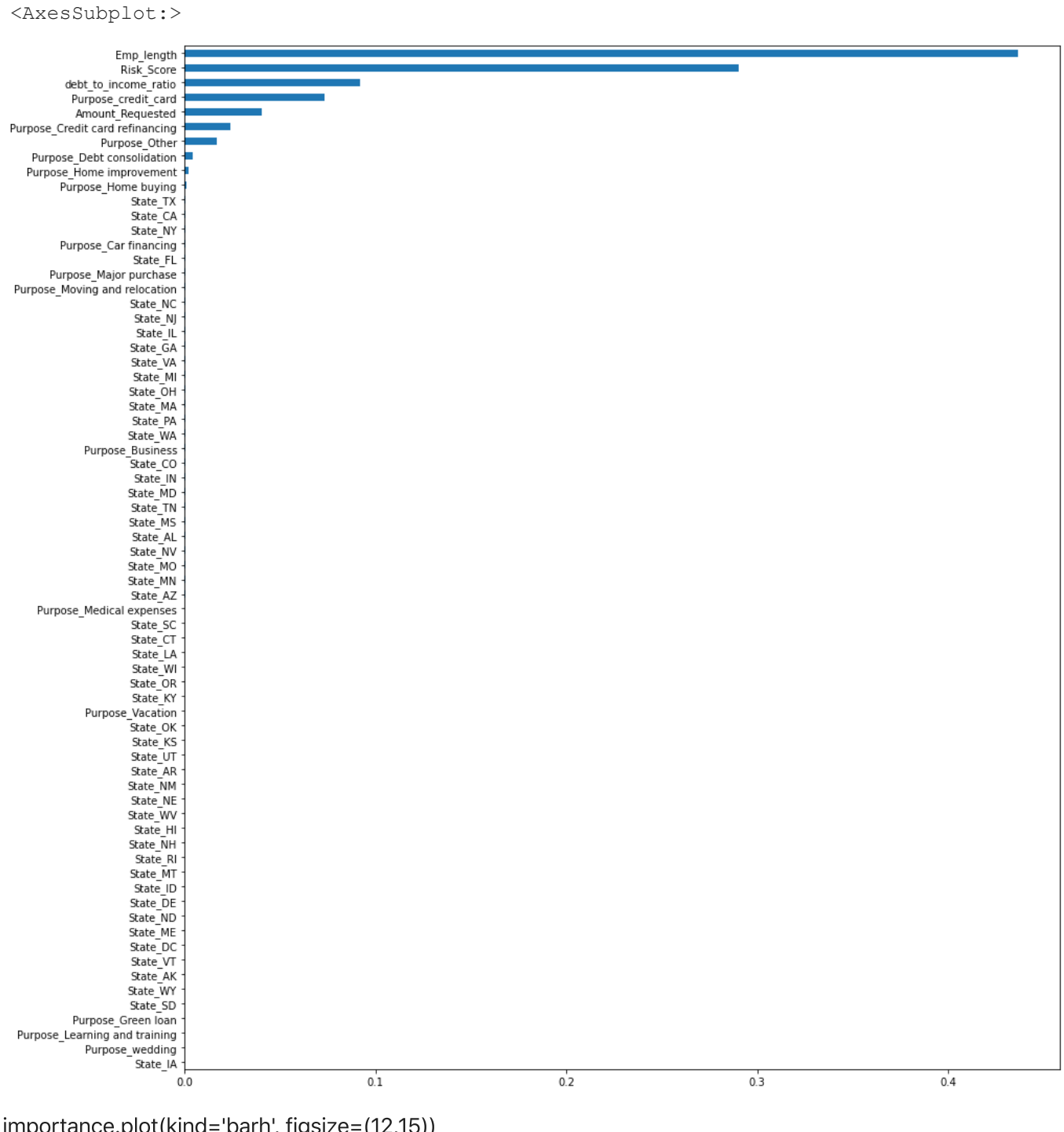
Out[68]:
```
GridSearchCV(estimator=RandomForestClassifier(n_estimators=10, random_state=42),
             param_grid={'max_depth': [1, 3, 5, 7, 9, 11, 13, 15, 17, 19],
                         'n_estimators': [1, 21, 41, 61, 81, 101, 121, 141, 161,
                                          181]},
             verbose=3)
```

In [98]:
```python
importance = pd.Series(rfc.feature_importances_,index=X.columns)
```

In [99]:
```python
importance = importance.sort_values()
```

In [100…
```python
importance.plot(kind='barh',figsize=(15,18))
```

Out[100…
```
<AxesSubplot:>
```



importance.plot(kind='barh', figsize=(12,15))

We can see the an applicant's employment history and their Risk score (credit score) are among the most important features.

Make a prediction with the testing class and evaluate

In [97]:
```python
# Make predictions with the hypertuned model
predictions = grid.predict(X_test)
```

In [94]:
```python
predictions
```

Out[94]:
```
array([1, 0, 1, ..., 0, 0, 0], dtype=int64)
```

In [95]:
```python
# Calculate classification report
from sklearn.metrics import classification_report
print(classification_report(y_test, predictions))
```

```
              precision    recall  f1-score   support

           0       0.95      0.98      0.97    145377
           1       0.98      0.94      0.96    113182

    accuracy                           0.96    258559
   macro avg       0.97      0.96      0.96    258559
weighted avg       0.96      0.96      0.96    258559
```

In [96]:
```python
# Calculate classification report - Add prediction labels
from sklearn.metrics import classification_report
print(classification_report(y_test, predictions,
                            target_names=["Rejected", "Accepted"]))
```

```
              precision    recall  f1-score   support

    Rejected       0.95      0.98      0.97    145377
    Accepted       0.98      0.94      0.96    113182

    accuracy                           0.96    258559
   macro avg       0.97      0.96      0.96    258559
weighted avg       0.96      0.96      0.96    258559
```

What this means

Precision - What percent of your predictions were correct?

Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class it is defined as the ratio of true positives to the sum of true and false positives. TP/(TP + FP)

Recall = What percent of the positive cases did you catch?

Recall is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives. TP/(TP+FN)

F1 score – What percent of positive predictions were correct?

The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. Generally speaking, F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy.