Loan repayment predictor: executive presentation

Alex Tyryshkin

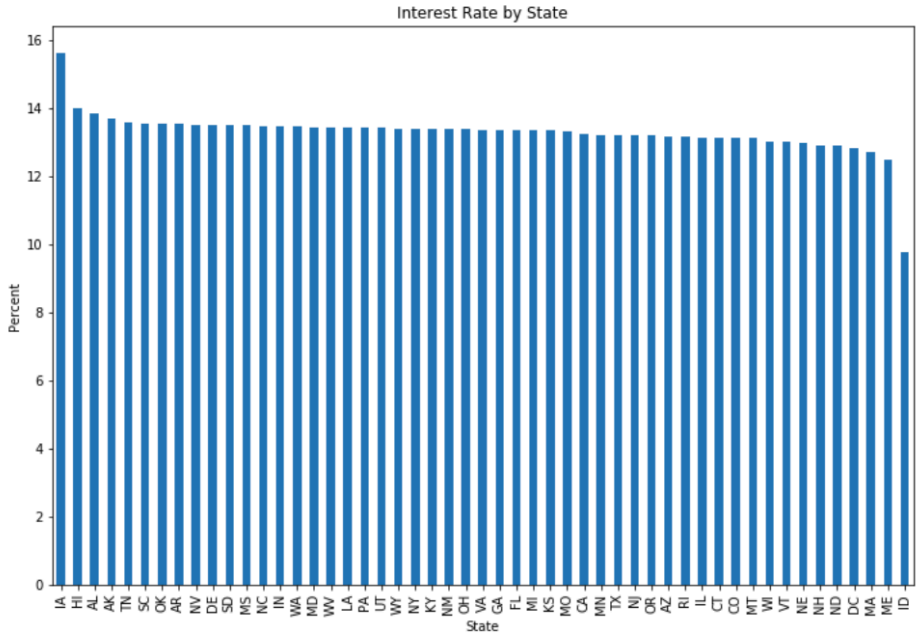
Data

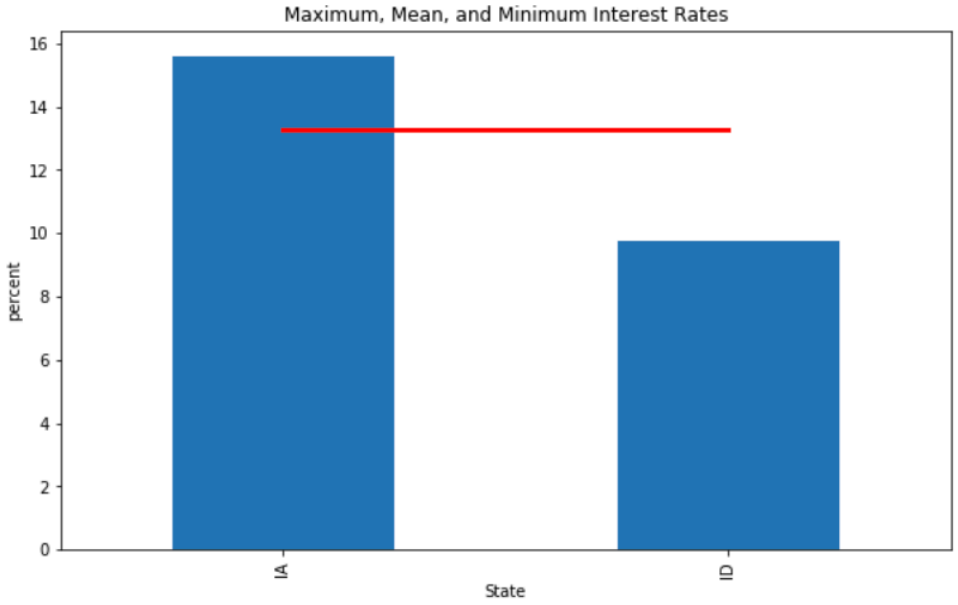
Sourcing:
lc_loan.csv is used as a starter file for training data
lc_2016_2017.csv is used as a starter file for testing data
https://www.kaggle.com/husainsb/lendingclub-issued-loans/notebooks

Processing:
Missing values removed
Dummy variables created
Training and testing datasets are equivalent in columns

EDA & Visualizations

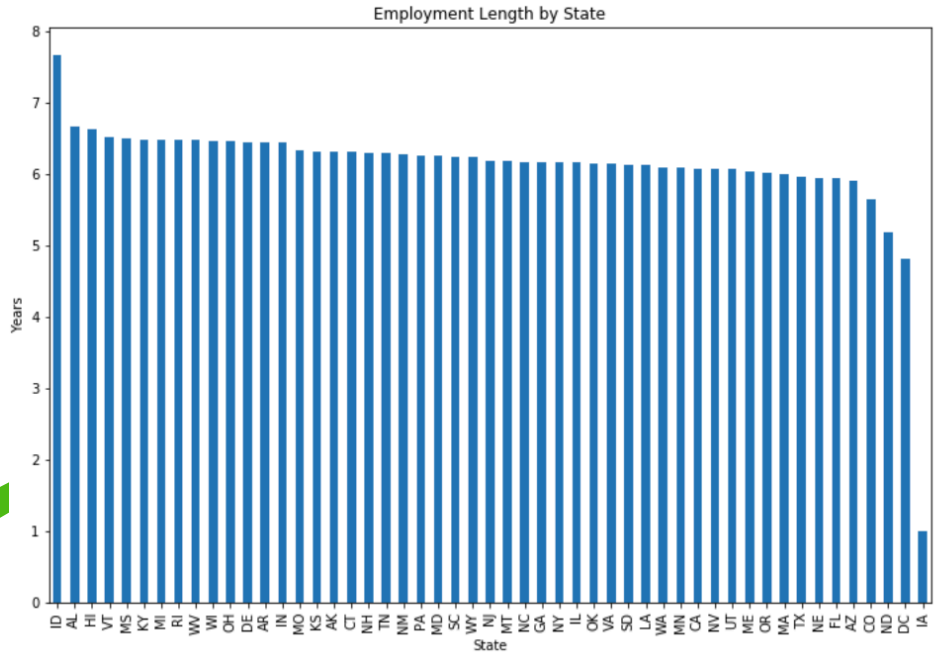
- Interest rates varied by states
- The state with the highest interest rate is IA and the lowest ID
- The state with the lowest interest rate is further away from the average interest rate than the state with the highest interest rate

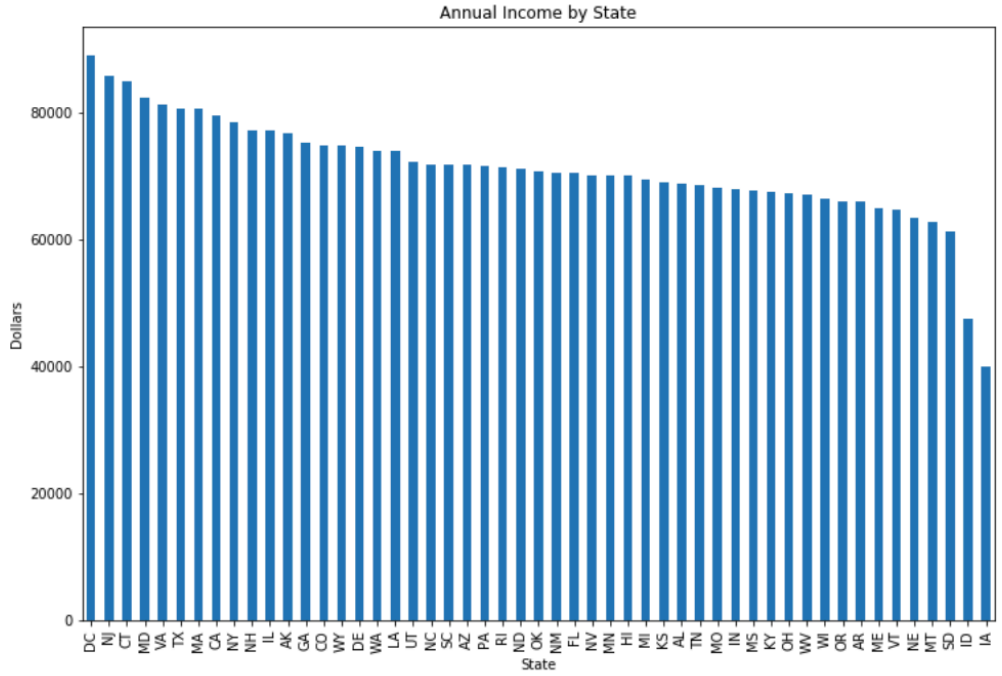




Employment & Annual Income by state

- The average employment length and annual income of borrowers varied widely across the states
- The states with the highest employment length are different from those with the highest average annual income

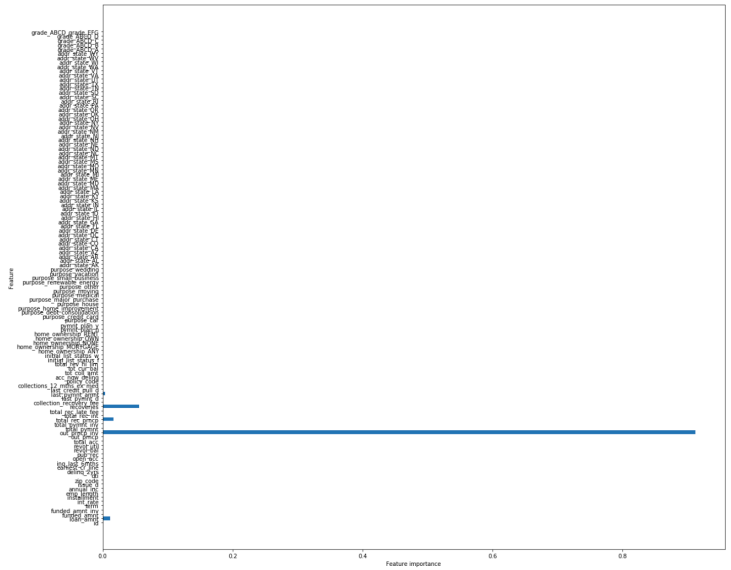




Modeling: Random Forest (initial)

-'out_prnpt_inv' and 'recoveries' were the most important features in our initial random forest model
- Grid Search determined the following parameters to be optimal:

```
In [13]: def plot_feature_importances(model):
             n_features = df_random_forest_train.shape[1]
             plt.figure(figsize=(20,18))
             plt.barh(range(n_features), model.feature_importances_, align='center')
             plt.yticks(np.arange(n_features), df_random_forest_train.columns.values)
             plt.xlabel('Feature importance')
             plt.ylabel('Feature')

         plot_feature_importances(tree_clf)
```




```
Grid Search found the following optimal parameters:
criterion: 'entropy'
max_depth: 6
min_samples_leaf: 10
min_samples_split: 5

Training Accuracy: 99.19%
Validation accuracy: 99.06%
```

```
# Confusion matrix and classification report
print(confusion_matrix(loan_status_current_test, pred))
print(classification_report(loan_status_current_test, pred)
```

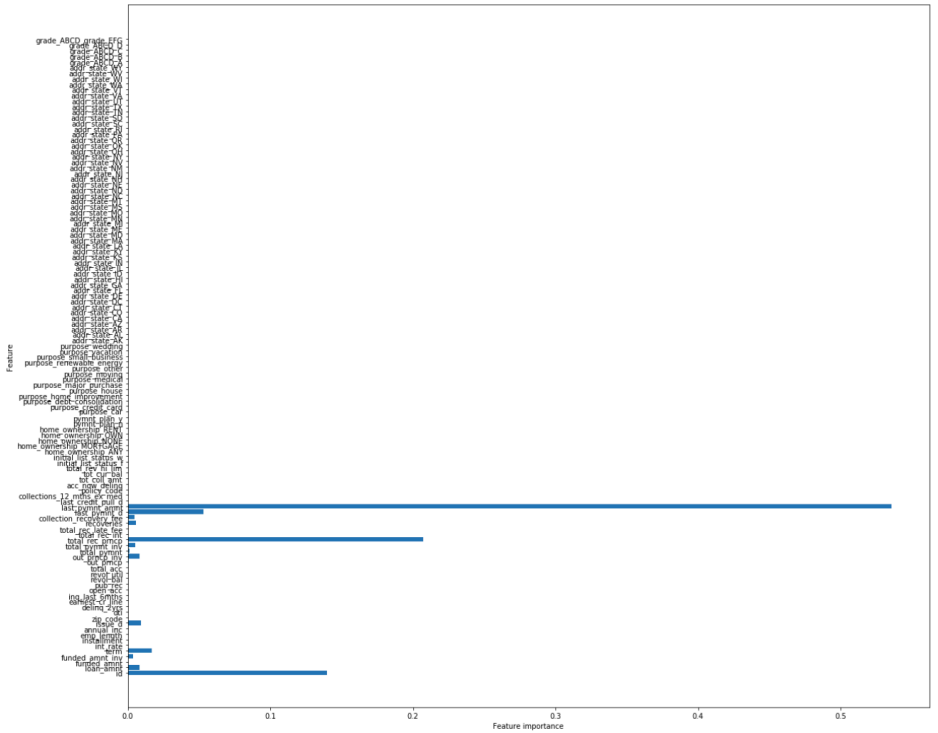
```
[[618615   9987]
 [     0 130718]]
              precision    recall  f1-score   support

           0       1.00      0.98      0.99    628602
           1       0.93      1.00      0.96    130718

    accuracy                           0.99    759320
   macro avg       0.96      0.99      0.98    759320
weighted avg       0.99      0.99      0.99    759320
```

```
print("Testing Accuracy for Decision Tree Classifier: {:.4}
```

```
Testing Accuracy for Decision Tree Classifier: 98.68%
```

Modeling: Random Forest (XGBoost parameters)

```
# Feature importance
plot_feature_importances(rf_tree_1)
```

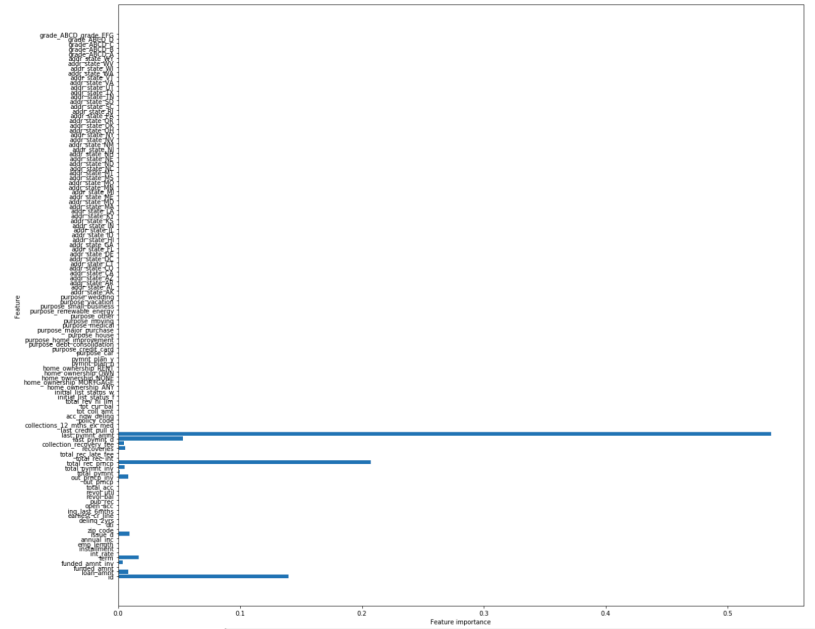



```
# Test set predictions
pred = forest_2.predict(df_random_forest_test)

# Confusion matrix and classification report
print(confusion_matrix(loan_status_current_test, pred))
print(classification_report(loan_status_current_test, pred))
```

```
[[626112   2490]
 [ 10850 119868]]
              precision    recall  f1-score   support

           0       0.98      1.00      0.99    628602
           1       0.98      0.92      0.95    130718

    accuracy                           0.98    759320
   macro avg       0.98      0.96      0.97    759320
weighted avg       0.98      0.98      0.98    759320
```

Modeling: Random Forest (after grid search)

```
# Feature importance
plot_feature_importances(rf_tree_1)
```



In [44]:
```
# Test set predictions
pred = forest_2.predict(df_random_forest_test)

# Confusion matrix and classification report
print(confusion_matrix(loan_status_current_test, pred))
print(classification_report(loan_status_current_test, pred))
```

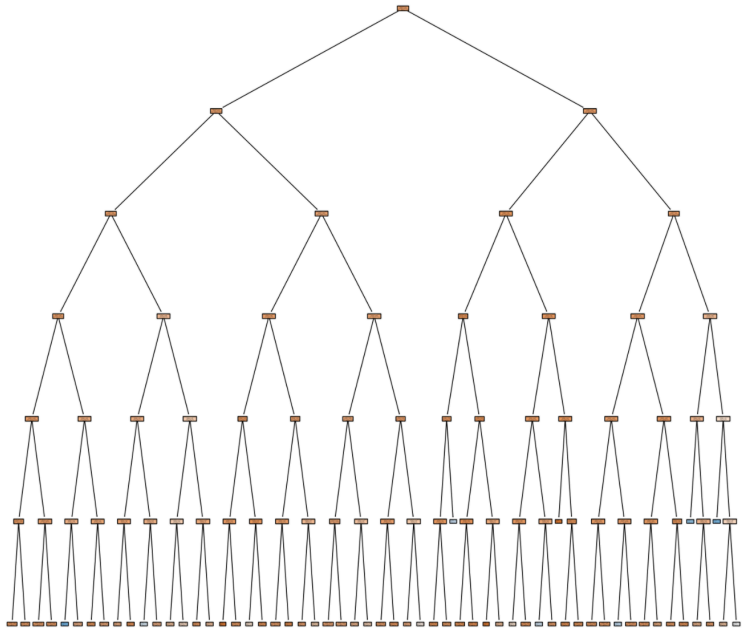
```
[[626112   2490]
 [ 10850 119868]]
              precision    recall  f1-score   support

           0       0.98      1.00      0.99    628602
           1       0.98      0.92      0.95    130718

    accuracy                           0.98    759320
   macro avg       0.98      0.96      0.97    759320
weighted avg       0.98      0.98      0.98    759320
```

Modeling: Decision Tree

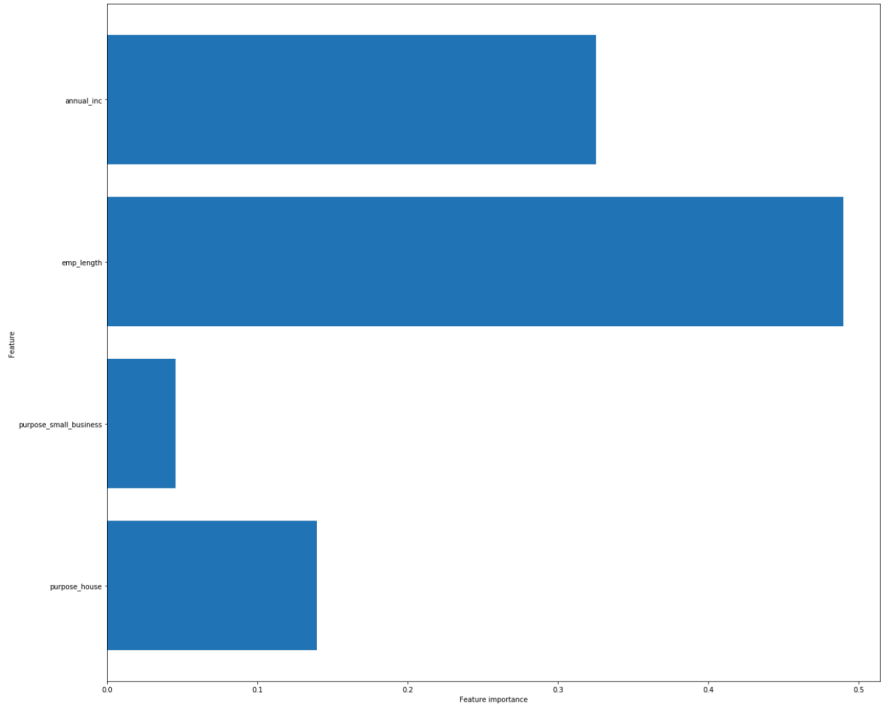
The Grid Search gave us the following optimal parameter

```
Grid Search found the following optimal parameters:
criterion: 'gini'
max_depth: 6
min_samples_leaf: 7
min_samples_split: 5

Training Accuracy: 79.22%
Validation accuracy: 82.78%
```

In [27]:
```
# Plot and show decision tree
plt.figure(figsize=(16,16), dpi=500)
tree.plot_tree(classifier_2,
               feature_names=X.columns,
               class_names=np.unique(y).astype('str'),
               filled=True, rounded=True)
plt.show()
```



In [37]:
```
## Plotting feature importances of decision tree classifier
def plot_feature_importances(model):
    n_features = X_train.shape[1]
    plt.figure(figsize=(20,18))
    plt.barh(range(n_features), model.feature_importances_, align='center')
    plt.yticks(np.arange(n_features), X_train.columns.values)
    plt.xlabel('Feature importance')
    plt.ylabel('Feature')

plot_feature_importances(classifier_2)
```



Summary

- The probability of default depended most heavily on annual income, employment length and whether the purpose was a home purchase

Further work

- Identify and clean more data related to the loan default process and create more combined feauteres
- Break down the geography by a group of state rather than individual state to explore other parameters in more depth

Thank you!