**Readme**

**Feature Selection**

* Dropped the unnecessary columns:
  + University Name
  + Rank
  + Mid-Career Pay
  + Degree Length
  + Region
  + State

University Name is the primary key for the table, Rank is state specific, and Mid-Career Pay is essentially another target variable so we elected to drop those columns right away. While we were initially interested in the difference of Early Career Pay between two-year and four-year degrees, we only had six rows of data associated with two-year degrees. Therefore, we did not use the Degree Length column either. After a few rounds of training the decision was made to also remove the Region and State columns as they may have been confusing the model and decreasing its performance.

The features we ended up keeping were the In-State and Out-of-State tuitions, Room and Board, Total Enrollment count, division, school type (public/private), Make World Better %, Stem %, and all of the diversity percentages.

**Data Preprocessing**

* Assessed the data types
* Encoded labels using Pandas.get\_dummies
* Replaced null values with zeros
* Converted continuous target values to categorical values

After feature selection we assessed the remaining data types and encoded the two object fields using Pandas.get\_dummies. We chose to replace null values with zeros instead of removing entire records. Out of the 907 rows, 29 were missing values for the Make World Better Percentage, 50 were missing for Room and Board, and 27 were missing the Enrollment and Diversity percentages. We elected to convert the target column values (Early Career Pay) to either “Low” (less than $45,000) or “Medium/High” for all other amounts using a lambda function. This was done to prepare the data for use with a classification model and to answer the question of whether we are able to accurately predict which features will result in a low early career pay.

**Model Selection and Benefits/Limitations**

Replace picture with text.

For our model we chose the Easy Ensemble Classifier using Adaptive Boosting.

Benefits

* Uses an ensemble of learners that evaluate previous errors and give those errors extra weight when fitting subsequent classifiers
* Utilizes random bootstrap sampling to help with overfitting
* Utilizes random undersampling to help with class imbalance

Limitations

* Outcome can be harder to interpret
* Slower to train and could significantly increase the amount of resources needed

**Model Training and Optimization**

Keep picture but remove first bullet point

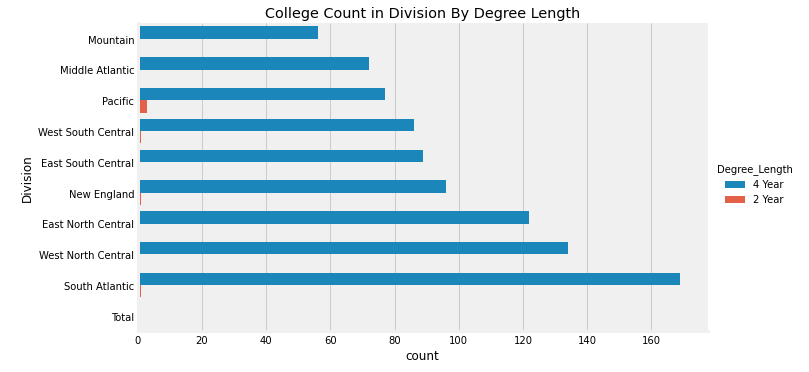
* Adjusted 75/25 train/test split to 80/20
* Increased the number of learners from 10 to 150
* Adjusted the sampling ration from 1.0 to 0.75

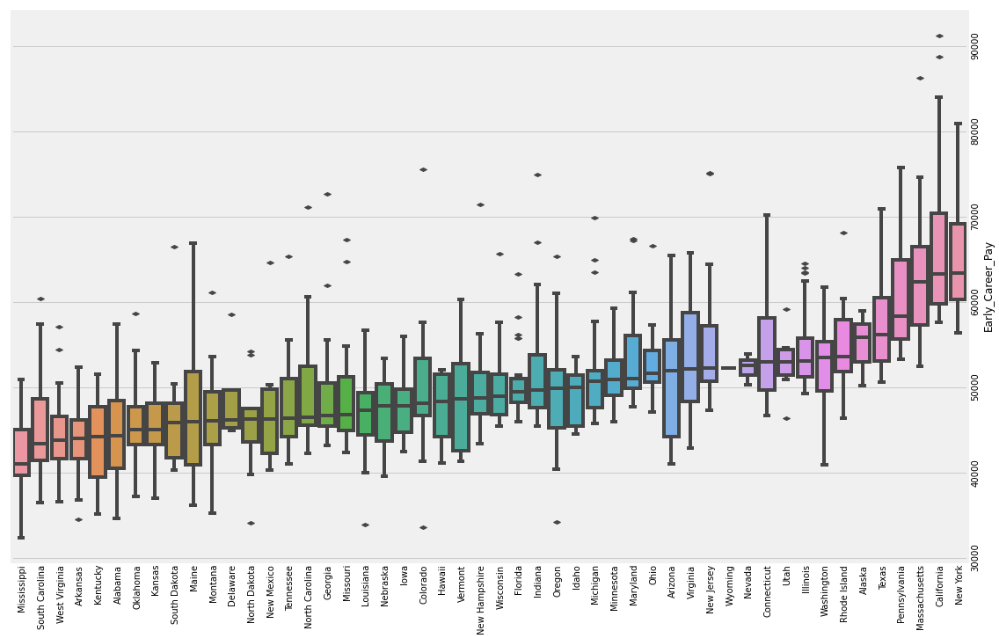
Our model is able to correctly predict 90% of the test targets. With a 97% recall rate, the model correctly identified 37 of the 38 early career salaries that are under $45K. There were 24 false positives (incorrectly predicted as low salary) resulting in a precision rate of 61%.

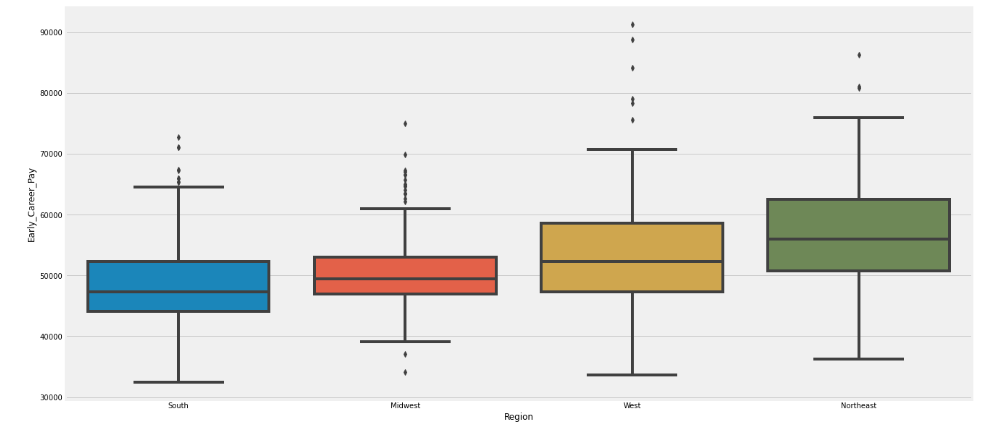
**Slides**

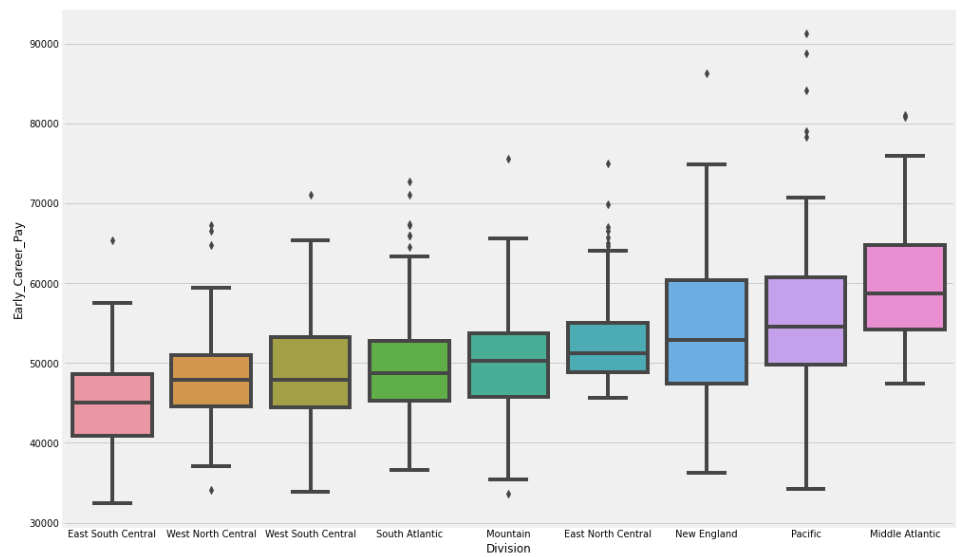
**Feature Selection**

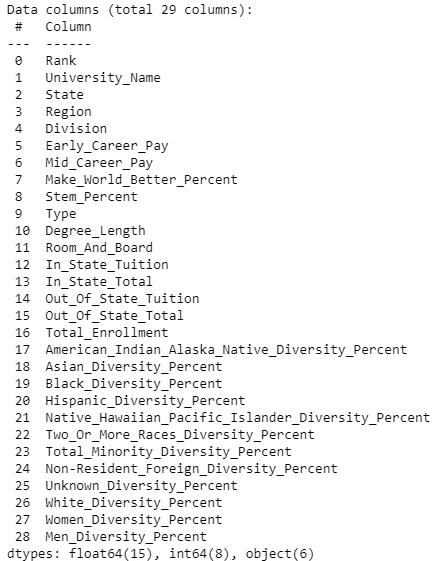
* Visual showing the count issue with the degree length field
* Visuals showing state/region vs division
* Visuals showing the original vs final features. Could maybe fomd something better for this?

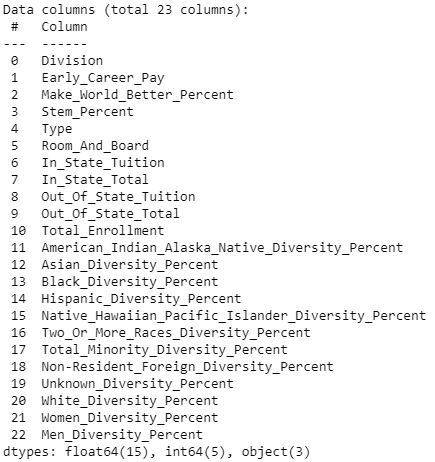












* Dropped columns:
  + University Name
  + Rank
  + Mid-Career Pay
  + Degree Length
  + Region
  + State

**Data Preprocessing**

* Visual showing preprocessing
* Any other visuals we could add to this slide?



* Assessed the data types
* Replaced null values with zeros
* Converted continuous target values to categorical values
* Encoded labels using Pandas.get\_dummies

**Model Selection and Benefits/Limitations**

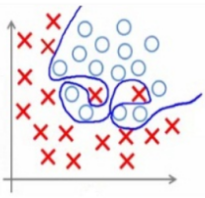
* EasyEnsemble visual/header
* Balancing/scale visual
* Overfitting visual
* Boosting visual



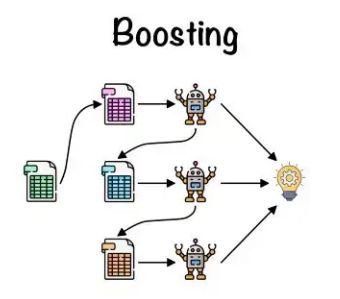
<https://imbalanced-learn.org/stable/references/generated/imblearn.ensemble.EasyEnsembleClassifier.html#ra96f85e96852-1>



<https://medium.com/@erica13chai/coding-problem-balancing-a-scale-11da8d88c823>



<https://moredvikas.wordpress.com/2018/09/24/over-fitting-in-context-of-machine-learning/>



<https://towardsdatascience.com/ensemble-learning-bagging-boosting-3098079e5422>

Benefits

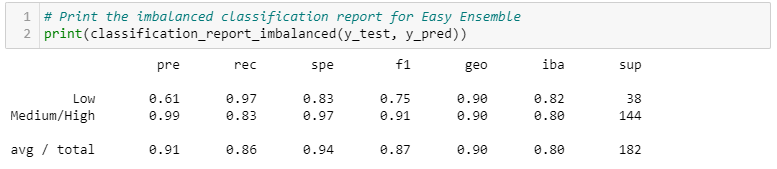
* Uses a bag of boosted learners that evaluate previous errors and give those errors extra weight when fitting subsequent classifiers
* Utilizes random bootstrap sampling to help with overfitting
* Utilizes random undersampling to help with class imbalance

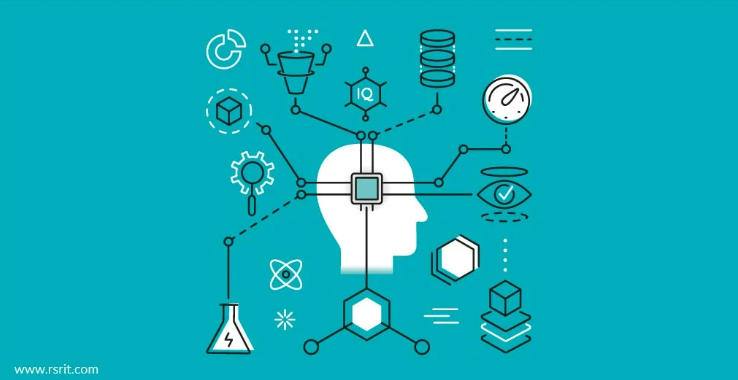
Limitations

* Outcome can be harder to interpret
* Slower to train and could significantly increase the amount of resources needed

**Model Training and Optimization**

* Add confusion matrix if possible, or
* Add an additional slide to show confusion matrix with some fun pictures?



****

<https://blog.rsrit.com/why-a-predictive-model-with-high-accuracy-might-not-be-a-good-model>

**Scripted Notes**

**Feature Selection**

Move slide 17 to be the first of the ML slides and highlight first 3 bullet points.

Of our 29 original columns, we ended up keeping 23 of them. The features we ended up keeping were the In-State and Out-of-State tuitions, Room and Board, Total Enrollment count, division, school type (public/private), Make World Better %, Stem %, and all of the diversity percentages.

The first 3 on our dropped list were dropped right away as the University name is our primary key for the table, Rank is state specific, and Mid-Career Pay is essentially another target variable.

Next slide…

Add dropped columns bullet points to right side of degree length slide and highlight 4th bullet point.

We were initially interested in the impact that two-year and four-year degrees make on Early-Career Pay, however, we noticed pretty quickly that only six rows of data were associated with two-year degrees. Therefore, we decided to drop that column as well.

Next slide…

Add another slide 17 before states slide with last two bullet points highlighted. Maybe consolidate 14, 15, and 16, with the bullet points in one quadrant. Words don’t need to be legible.

After a few rounds of training the decision was made to also remove the Region and State columns as they may have been confusing the model and decreasing its performance. The State column grouped the data into 50 categories, Region grouped by 4, and Division by 9 so using the Division column instead of State and Region appeared to be a nice middle ground that worked out well for our model.

Next slide…

**Data Preprocessing**

After deciding which features were important, we worked through getting the rest of the data ready for our model. For our first step we replaced all null values with zeroes. Out the 907 records in our dataset, there were 29 missing values for the Make World Better Percentage, 50 for Room and Board, and 27 for the Enrollment and Diversity Percentages.  We chose to replace with zeros instead of removing the records entirely.

The next step of our pre-processing involved using a lambda function to convert our target column amounts into either "Low" or "Medium/High".  Any amount that was under $45K was converted to “Low” and everything else to “Medium/High”. This was done to prepare the data for use with a classification model and to answer the question of whether we are able to accurately predict which features will result in a low early career pay.

And finally, we assessed the remaining data types and encoded our two object fields (Division and Type) using Pandas get\_dummies.

Next slide…

**Model Selection and Benefits/Limitations**

For our model we chose the Easy Ensemble Classifier using Adaptive Boosting. We chose this one because of its ability to give higher weights to errors during the learning process and because it utilizes random bootstrap samples and under-sampling to help with overfitting and class imbalance. However, a couple limitations with this classifier are that the outcome can be harder to interpret and it can be slower to train than other algorithms.

Next slide…

**Model Training and Optimization**

This table shows the different steps that were taken to optimize our model. They didn’t necessarily happen in this order but it’s a good visual showing the increasing accuracy scores that brought us to our final score of 90%.

The first row can be considered our starting point after we had dropped 4 features. This was using the default settings of a 75/25 train/test split with 100 AdaBoost learners and a sampling ration of 1.

We then dropped the State and Region features. After that we updated the train/ test split to 80/20. Then we increased the AdaBoost learners to 150. And finally, we changed the sampling strategy from 1 to 0.75 which brought us to our accuracy score of 90%.

Next slide…

Our confusion matrix and classification report show that the model correctly identified 37 of the 38 early career salaries that are under $45K resulting in a recall of 97%. We ended up with a precision rate of 61% for the Low category, meaning there were 24 schools incorrectly predicted as having a low early career salary. Overall, we were more concerned with being able to identify ALL low pay schools so recall was more important to us than precision.

**Outline**

**Feature Engineering**

dropped columns and why

picture of the region EDA - with division and state?

what features were included?

**Pre-process**

Assessed data types and encoded object labels with Pandas get\_dummies

replace nulls with zeros

      why?  didn't remove records

      which columns had nulls?

Convert target column in order to use a classification model - lambda function

**Model choice**

benefits/limitations

      slower to train

      harder to see how the model works

      uses bootstrapping to overcome overfitting problem

how was model trained/optimized?

      trained 725, tested 182

      number of learners in the ensemble

      sampling ratio = 0.75

confusion matrix and accuracy score

Not used:



