Justin Dagenhart
Muhammad Adil Sohail
Shivam Mishra
Corbyn Yhap

ENPM 808W

Purpose:

The process of applying for a United States Work Visa can be time consuming, and costly. On top of that, there is not a guarantee that an applicant will be accepted. Our team decided to try and tackle this problem by attempting to create a classifier that would ascertain whether an individual will be certified or denied their work visa. If an individual wants to do a check to try and save them some time, and money before applying, the idea is that they could potentially try our classifier.

What We Did:

Steps taken:

1. Dataset from Kaggle was processed using the python script, “InitialCleanUp.py”
2. Data was then further cleaned using the “InitialCleanUp_2.R” script (Eliminated fields we didn’t care about
3. The output of step 2 is then fed through our classifier python scripts, “ComparisionClassifier.ipynb”, and “FinalClassifier.ipynb”
4. The rest of our script files were used for analysis (on data after step 2)

Exploratory Analysis:

The initial dataset we found on kaggle can be found here:

https://www.kaggle.com/jboysen/us-perm-visas

This dataset contained 205,533 observations scrubbed from the Department of Labor’s website between the years 2006 to 2015. It contained over 154 fields. At 285Mb, it ended up being a lot of data to work with so we had to figure out what fields we wanted to keep and which to throw out. We did the following to clean the data:

- We started out by throwing out features that contained primarily no data (this cleaned up a large portion of the fields)
- We also had to combine a few fields that were repeats (country_of_citzenship and country_of_citizenship and others) These fields had no overlaps so they could just be combined into one feature
- After removing fields this left us with a much smaller choice selection, We Chose features we thought would make good features that would help in our classifier

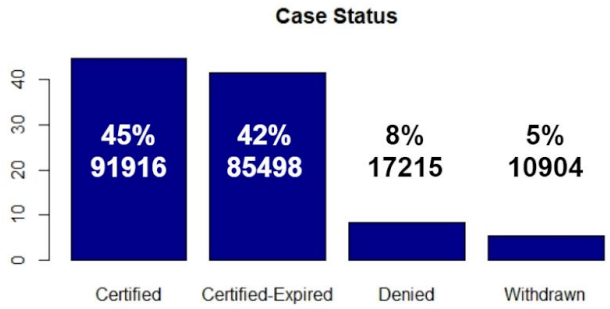
-

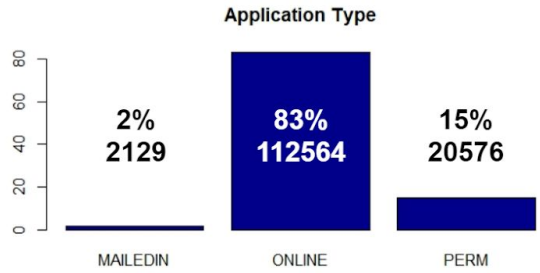
Features we Ended Up keeping for Analysis or Classification:

Features	Description	Notes
"case_no"	Unique identifier per entry	Contained data identical to another field that we combined (case_number)
"case_status"	Status associated with the last significant event or decision. Valid values include “Certified,” “Certified-Expired,” “Denied,” and “Withdrawn”	This field is the information we wanted to try to guess correctly using a classifier
"agent_state"	State information for the Agent or Attorney requesting a permanent labor certification on behalf of the employer.	State data was not all in the same format (Maryland Vs. MD), data was formatted such that state data was homogenous in format (all lower case abbreviations)
"case_received_date_EPOCH"	Date the application was received by the ETA National Processing Center (Time since the Epoch) We created this new variable from the original time format which we had to parse	Case received date was given as YEAR-MONTH-DAY, this was converted to time since the epoch to be more usable
"case_received_date_YEAR"	Date the application was received by the ETA National Processing Center (Only the Year) We created this new variable from the original time format which we had to parse	Same processing done as above except the year value was just extracted
"class_of_admission"	Indicates the class of immigration visa the foreign worker held at the time the permanent labor certification application was submitted for processing (if applicable)	There were 57 unique entries here. ~80% of the data was H-1B visas, so we just grouped in H-1B or Not H-1B
"country_of_citizenship"	Country of citizenship of the foreign worker	Made all values lowercase
"employer_state"	State of the employer requesting permanent labor certification	Same problem as in agent_state
"application_type"	Application type submitted	

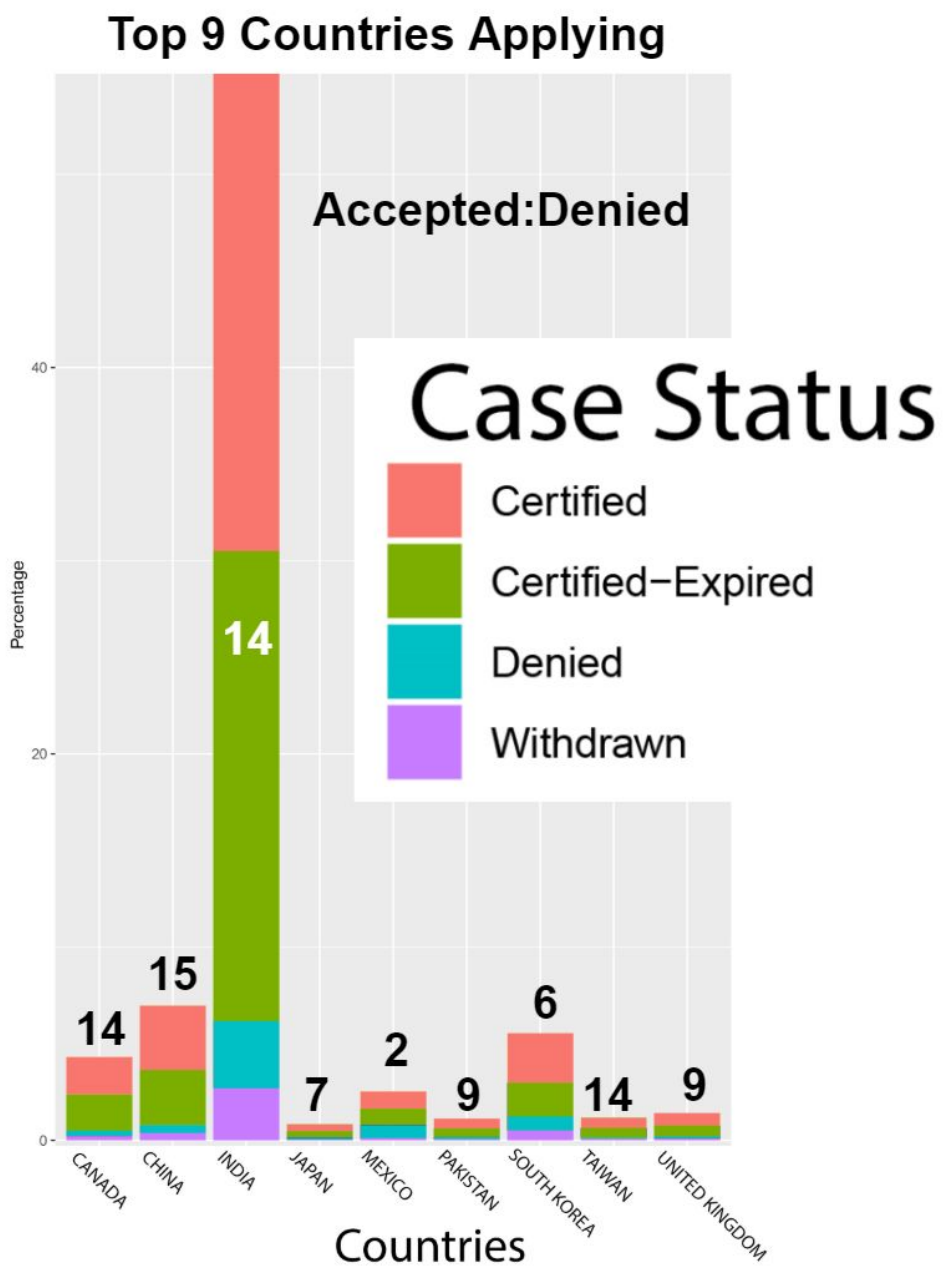
	(Perm/Online/Mailedin)	
"decision_date_YEAR"	Date on which the last significant event or decision was recorded by the ETA National Processing Center (Year only) We created this new variable from the original time format which we had to parse	Same processing as was done in case_received_date but was only kept the YEAR data as exploratory analysis
"employer_postal_code"	Zip Code of the employer requesting permanent labor certification	
"employer_name"	Name of the employer	Used only for data Analysis
"foreign_worker_info_education"	Highest Education achieved by the foreign worker	
"job_info_major"	Major field of study required based on the education requirement	
"pw_amount_9089"	Prevailing wage for the job being requested for permanent labor certification	This value was scaled to be annual salary using the associated unit_of_pay field
"pw_unit_of_pay_9089"	Unit of Pay. Valid values include “Hourly (hr)”, “Weekly (wk),” “BiWeekly (bi),” “Monthly (mth),” and “Yearly (yr)”	Only used to convert pw_amount_9089 to annual salary
"us_economic_sector"	US economic sector that the job resides in	
"wage_offer_from_9089"	Lower range of the wage offer	This value was scaled to be annual wage offered using the associated unit_of_pay field
"wage_offer_unit_of_pay_9089"	Unit of Pay. Valid values include “Hourly (hr)”, “Weekly (wk),” “BiWeekly (bi),” “Monthly (mth),” and “Yearly (yr)”	Only used to convert wage_offer_from_9089 to annual wage offered

After extracting only the data we wanted, we plotted various things in order to get an idea of how the data looked.

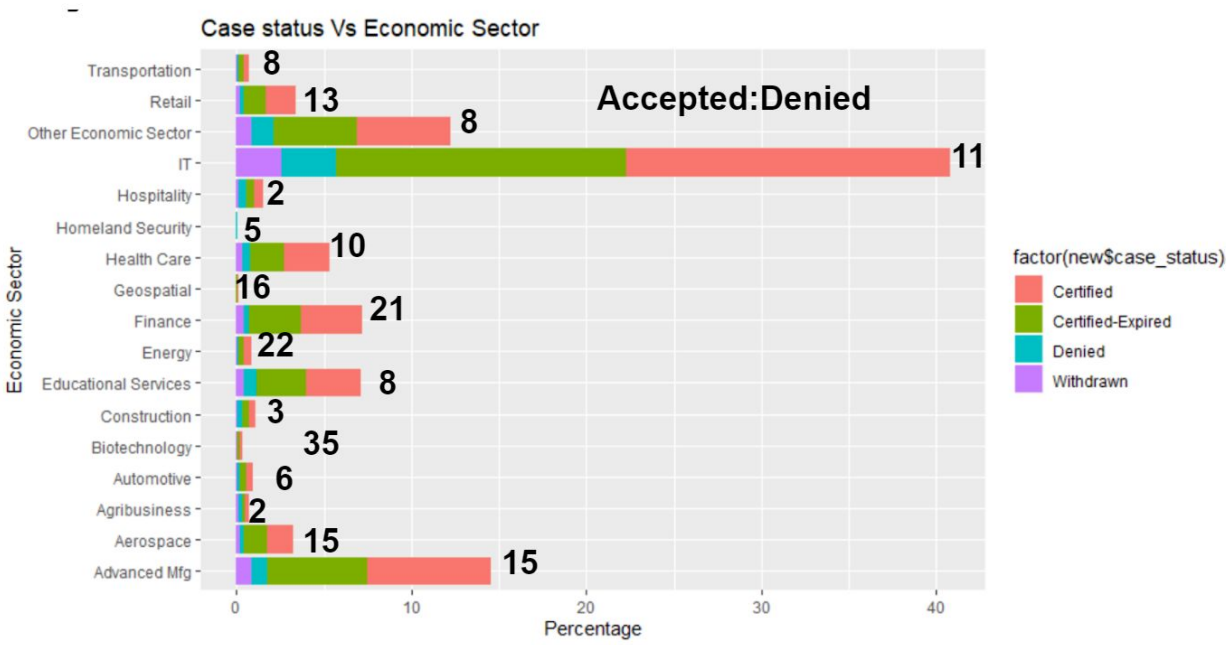




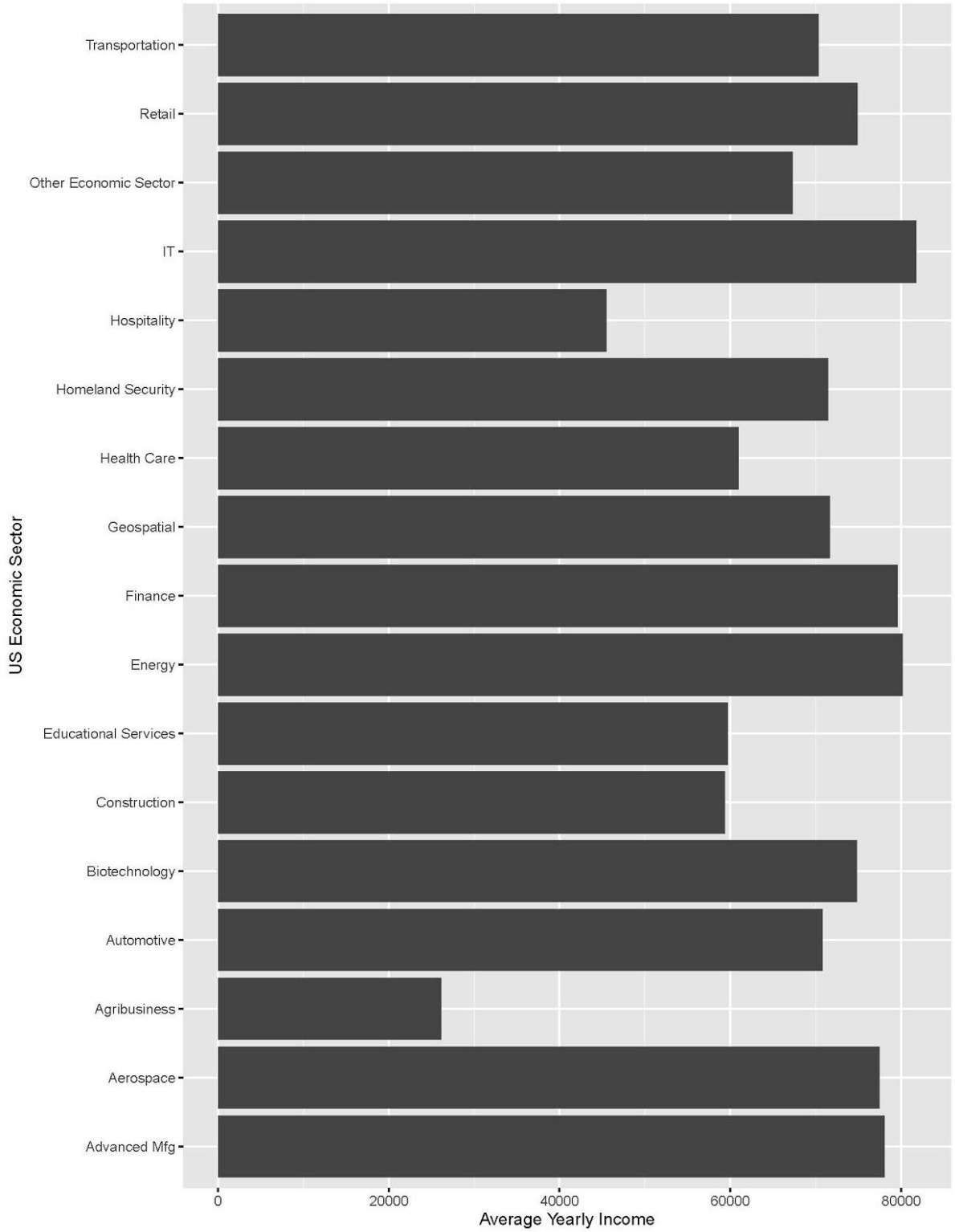
Most of the application status entries in our data were Certified, or Certified Expired for Case Status and most of our visa applicants submitted their applications online. The ratio of Accepted (Certified and Certified-Expired) to Denied (only Denied) is 12:1.



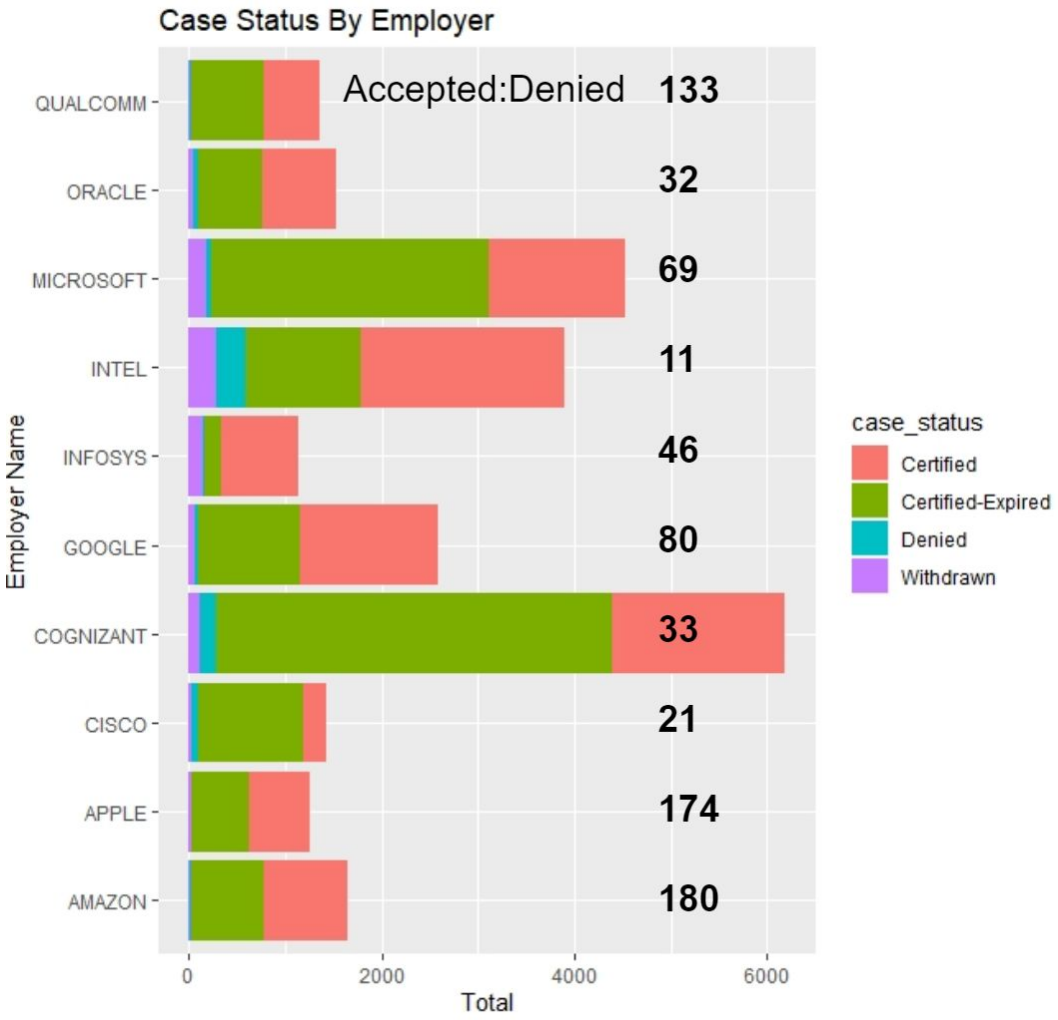
Surprisingly, many of the applicants received in our data were predominantly Indian. (we were expecting a larger percentage of Chinese and other countries) Among the top 9 countries applying, China had the largest Accepted Vs. Denied ratio and Mexico had the lowest.



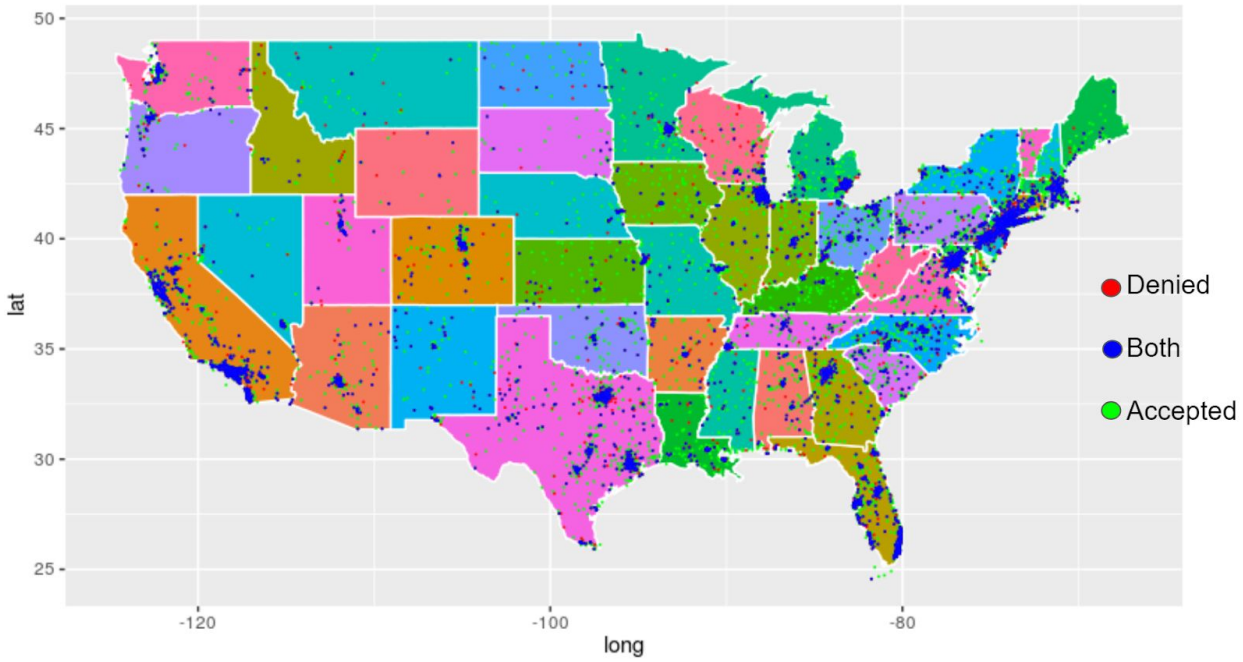
Not surprisingly Information Technology sector had the most applicants apply and Advanced Manufacturing at second. The US Economic Sector with the highest Accepted Vs denied ratio was Biotechnology and the lowest was Hospitality, and Agribusiness.



Average annual wage for visa applicants in IT, Finance, Energy, Advanced Manufacturing, and Aerospace are all fairly close. Lowest average annual wage was in Agribusiness and Hospitality



Employers with the highest number of visa applicants were Cognizant Technology Solutions and Microsoft in second. The company with the highest Accepted Vs. Denied ratio was Amazon with Apple close behind and Intel had the lowest.



We also were curious as to where the companies were that applicants were applying to so we plotted a map of the USA with dots representing employer locations. You can see the majority of applicants applying to employers along the East and West coasts. With additional clusters of applicant companies at more populated cities in the US. No discernable pattern was found for Accepted Vs Denied for location (no effect by itself).

Classifier Creation and Results:

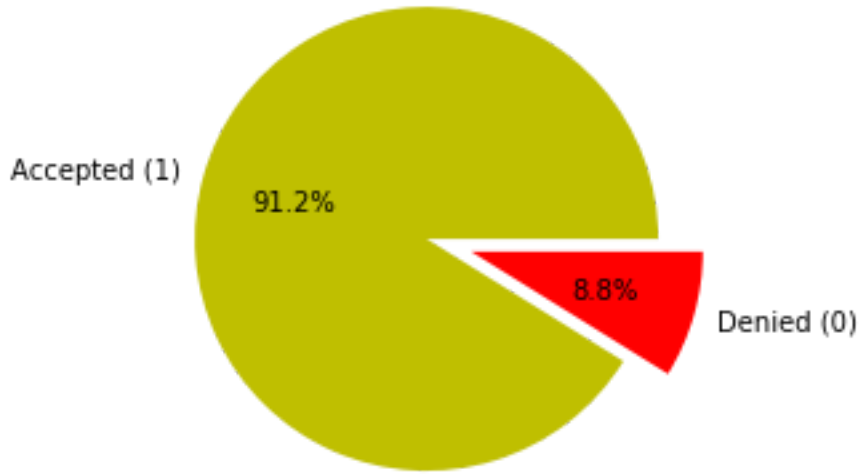
Case status for all the rows in the dataset is divided into three cases, namely:

1) Accepted
2) Denied
3) Withdrawn

The data rows with a withdrawn case status were dropped from the data-set. The withdrawn case status stands for around 5% of the whole data-set. The remaining data rows have only two values for the case status, accepted and denied. **Accepted** case status is denoted by integer 1 in the data-set, while **denied** case status is denoted by the integer 0.

The case status for more than 90% of the data rows is accepted while the denied cases account for less than 10% of the total cases. This is portrayed in the figure below.

The data-set is highly imbalanced in this respect. Another major issue with the data-set is the missing values. There was not a single row in the original data set that carried all of the features. The original dataset contained more than 100 data features, and plenty of them were redundant or of no use for the classification of accepted and denied cases.



This final classifier we settled on extracted eight features from the data-set along with the case status feature. The case status is the dependent variable, this classification model attempts to predict the case status of the rows based on the other eight independent features. The table below lists the eight independent features used for classification, along with the percentage of data-rows that have these features missing in the original dataset.

Data Features	Percentage of missing data
1. Case received date YEAR	65.8%
2. Application type	34.2%
3. Class of admission	5%
4. Education level	65%

5. Country of citizenship	0%
6. Economic sector	37.2%
7. Employer state	0%
8. CTC (Cost to Company)	0%

All the features except CTC are categorical in nature. The missing cases for these features were dealt with by treating missing data values as another category. A missing category was added to the existing categories of the above data features.

Using this modified data-set, a Decision Tree Classifier was trained on 70 percent of the original data-set, and tested on the 30% of the original data-set. As described above, the case status label of the data-set is highly imbalanced. The accuracy results on the test-set is summarized in the table below.

Percentage of **denied** cases predicted correctly.	Percentage of cases predicted correct. (Accuracy Score)
7.06%	93.72%

The accuracy score for the data is very high, approximately equal to 94% but the classifier is not able to predict the denied case status in the test dataset very well. Specifically, it was able to accurately predict only 7 percent of the denied cases in the test data-set. This suggests that the classifier is predicting accepted case status for almost all of the data rows present in the test set, and since more than 90% of the data-set contains accepted case status values (and hence the test data set also contains the majority of data rows with accepted case status values), the accuracy is high.

Let's now fix the test data-set in such a way that one can get a clear picture of the classifier accuracy, not some biased view as we got above. This is done here by creating a test data set that contains 3017 denied cases and 3000 accepted cases (test case contains almost equal amount of both the accepted and denied cases). This test data-set will be referred as TDS dataset from this point on. The performance of the classifier trained on all the data and tested on the TDS dataset is summarized in the tables below.

Percentage of **denied** cases predicted correctly.	Percentage of cases predicted correct. (Accuracy Score)
8.51%	54.03%

	Predicted Denied	Predicted Accepted
Actual Denied	257	2770
Actual Accepted	6	2994

The confusion matrix for the classifier trained on the whole data-set confirms the inference that the classifier is predicting accepted case status for almost all the data rows in the TDS data-set. This is expected due to the fact that the model has been trained on a data-set that is highly skewed in favor of accepted case status.

We tackled this problem by under sampling the data-set. Under sampling is the removal of data rows belonging to the majority class. As the original data-set is very large, it can be expected that even after under-sampling the majority class, the final data-set would be sufficient to capture its necessary variations. Using the modified data-set, different classifiers are used for predicting case status. The table below summarizes the performance of each classifier on the test data-set*.

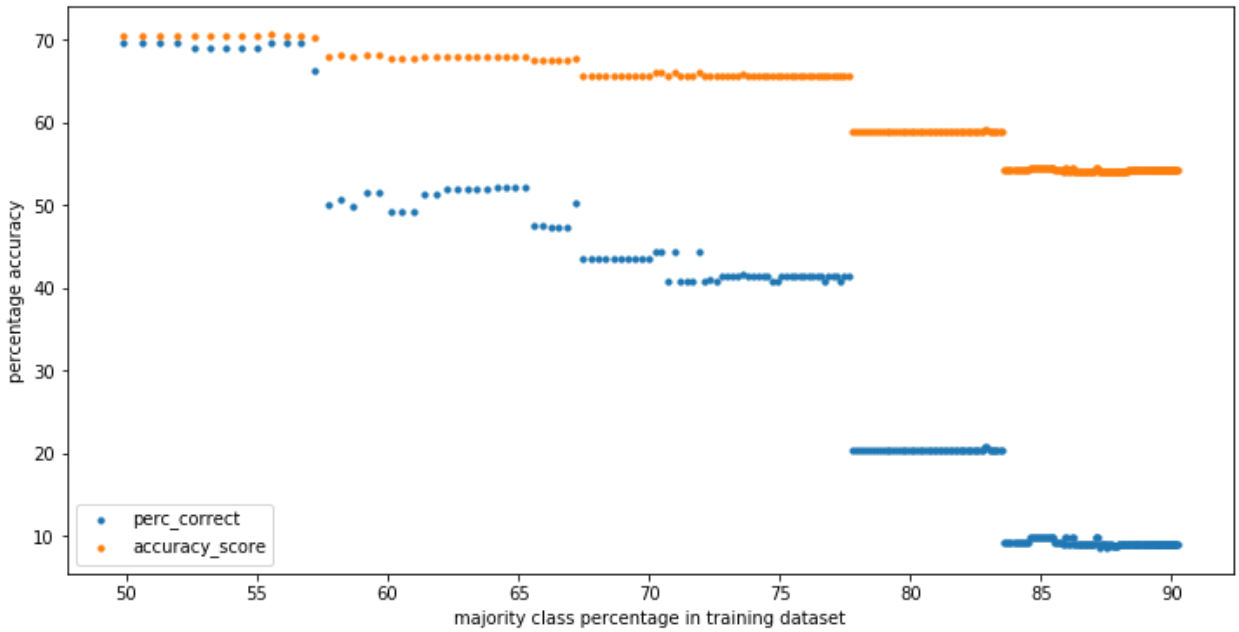
*the test data-set for comparison is different from the TDS used everywhere else.

Classifier	**Accuracy**	
	Perc accuracy	accuracy score
Logistic Regression	48.62	50.24
Support Vector Classifier	49.12	50.74
Random Forest Classifier	**71.67**	**71.74**
Decision Tree Classifier	**73.88**	**70.12**

Tree based classifiers clearly outperform the Logistic Regression and the Support vector classifier. Also, it can be noted that the Random Forest and the Decision Tree classifier performs equally well. The accuracy score for the decision tree is slightly better than the Random Forest. The models above are trained on the dataset that contains 7000 accepted cases, and 7038 denied cases (approximately 50% mixture). The accepted cases for training are selected at random from the original data-set.

The below graph represents the variation of the accuracy score and perc_correct score (percentage of denied cases correctly predicted) on the TDS data-set. The accuracy score, decreases overall with the increase in the majority class percentage. The value however remains constant for a range of majority class percentage before depreciating.

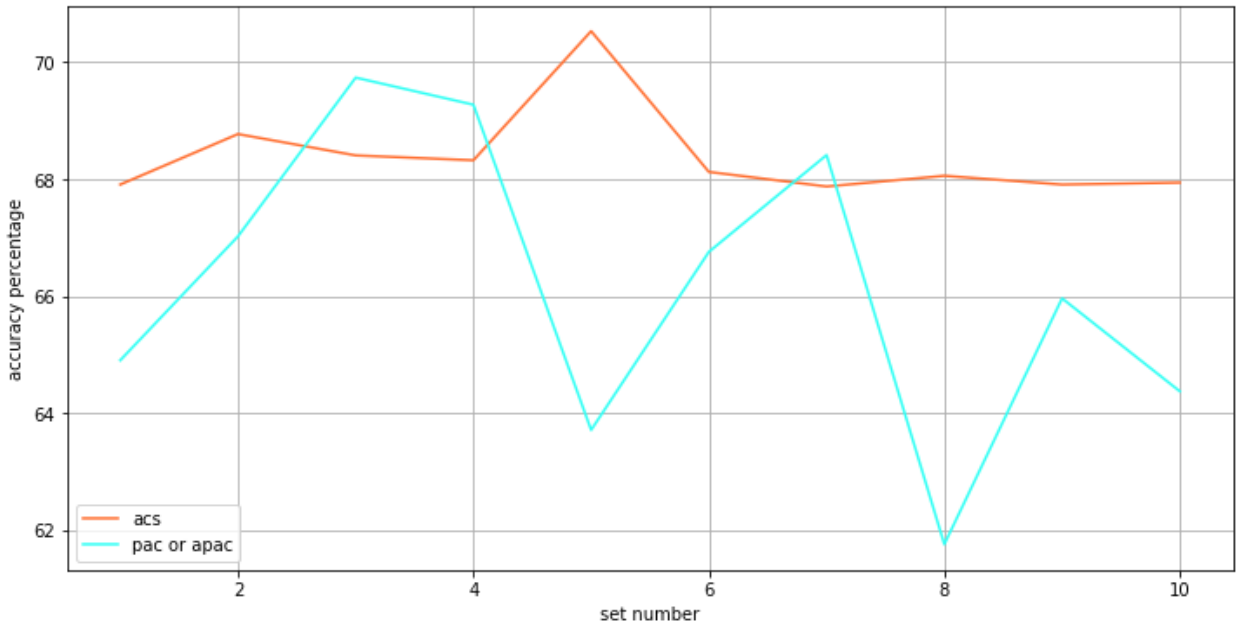
The perc_correct score remains constant at around 70 % before dropping to a significantly low percentage once majority class in the training dataset exceeds 57% and keeps on dropping with the increase in the majority class percentage.



Above reported accuracy of the Decision Tree Classifier is based upon the training of the model on a dataset containing 7039 denied cases and 7000 accepted cases. The accepted cases are selected randomly from the big set of accepted cases data rows. As, these are selected randomly from all over the data-set, it can be intuitively concluded that they capture all the variations for the accepted cases data rows in the original dataset.

The original dataset contains around 70000 data rows with the accepted case status. In the next part the decision tree is trained on ten unique sets of accepted data rows which were segmented randomly from the 70000 accepted data rows mentioned above. The accuracy of the models trained is checked on the TDS data-set. The results for the same are summarized below.

Set Number	Accepted Predicted As Accepted	Denied Predicted As Denied
1	67.9	64.89
2	68.77	67.02
3	68.4	**69.74**
4	68.32	69.27
5	**70.533**	63.7
6	68.12	66.7
7	67.87	68.41
8	68.05	61.75
9	67.90	65.95
10	67.94	64.36
Average	**68.3**	**66.18**



The accuracy score remains almost the same for all the sets. The perc correct score (percentage of denied cases correctly predicted) on the other hand varies a little. The average for both is greater than 65 percent.

Who Did What?

Adil:

- Initial Data Pruning
- Plots
- Presentation

Corbyn:

- Plots
- Presentation
- General review of classifier

Justin:

- Initial Data Cleaning
- Write Up
- Presentation

Shivam:

- Final Classification code (Python)
- Write Up (classifier section)
- Presentation (classifier section)

Did Our Technique Work?

Using the initial data set which carried a majority of accepted cases, our classifier was unable to have a better accuracy than if you were to just randomly guess accepted visa for every single data point. In that regard, we were unable to do better than the typical baseline used in many classification problems seeing as a human could achieve the “guess only achieved” accuracy of over 90%. Our data was extremely skewed, so we instead would rather think about how accurately were we able to properly classify denied cases. If we went with assuming all cases are accepted, we would get a 0% accuracy with this naive classifier. However with our best classifier we were able to train, we were able to get both our accuracy for denied and accepted cases up to ~70% each. So as a classifier trying to predict whether a visa applicant will be denied, we believe that we would be able to tell 70% of applicants whether or not their application would be denied. This is sufficient as it would aid both applicants and companies a lot of money when deciding whether or not to apply.