A PROJECT ON

"H1B VISA APPROVAL PREDICTION"

SUBMITTED IN
PARTIAL FULFILMENT OF THE
REQUIREMENT FOR THE COURSE
OF POST GRADUATE DIPLOMA IN
BIG DATA ANALYTICS

SUNBEAM INSTITUTE OF INFORMATION TECHNOLOGY

Kasarsai Rd, Phase 2, Hinjewadi Rajiv Gandhi Infotech Park, Hinjewadi,
Pimpri-Chinchwad, Maharashtra 411057

SUBMITTED BY:
Pratik Kanade (69642)

UNDER THE GUIDANCE OF:
Mrs. Manisha Hingne
Faculty Member
Sunbeam Institute of Information Technology, Pune.

SUNBEAM

CERTIFICATE

This is to certify that the project work under the title ‘H1B Visa Approval Prediction’ is done by Mr. Pratik Kanade in partial fulfilment of the requirement for award of Post Graduate Diploma in Big Data Analytics Course.

Mrs. Manisha Hingne — **Project Guide**

Mrs. Pradnya Dindorkar — **Course Coordinator**

Date : 3/13/2023

ACKNOWLEDGEMENT

A project usually falls short of its expectation unless aided and guided by the right people at the right time. We avail this opportunity to express our deep sense of gratitude towards Mr. Nitin Kudale (Center Coordinator, SIIT, Pune) and Mrs. Pradnya Dindorikar (Course Coordinator, SIIT ,Pune) and Project Guide Mrs. Manisha Hingne.

We are deeply indebted and grateful to them for their guidance, encouragement and deep concern for our project. Without their critical evaluation and suggestions at every stage of the project, this project could never have reached its present form.

Last but not the least we thank the entire faculty and the staff members of Sunbeam Institute of Information Technology, Pune for their support.

Pratik Kanade
DBDA September 2022 Batch,
SIIT Pune

TABLE OF CONTENTS

1. Introduction

1.1 Introduction

H1B visa is a type of non-immigrant visa that allows foreign workers to work in the United States for a specific period of time. To be eligible for an H1B visa, the applicant must have a job offer from a U.S. employer, have a bachelor's degree or higher in a specialised field, and the job offered must require the applicant's specialised knowledge and expertise. There is an annual cap on the number of H1B visas that can be issued each year. For the fiscal year 2022, the cap is 85,000. The employer must file a petition as the Labor Condition Application (LCA) with the U.S. Department of Labor on behalf of the applicant as the first step in the process. In the second step, the approved visas are then filed with the U.S. Citizenship and Immigration Services (USCIS). If the petition is approved, the applicant can then apply for the H1B visa at a U.S. consulate or embassy. H1B visa holders can have dual intent, meaning they can apply for permanent residency (green card) while on the H1B visa. Overall, the H1B visa program allows U.S. employers to hire foreign workers with specialised knowledge and skills to fill job vacancies.

1.2 Objective

The main objective of this project is to predict the probability of approval of the US work visa for non immigrant workers settled in the United States of America. The machine learning model will also verify if the H1B is granted on a random basis to the employees, which is considered to be the case, or is dependent on different features. Along with data analytics, required data analysis of various factors that might contribute towards granting of visa is conducted. A web application to predict the approval of visa is developed using streamlit in python.

1.3 Why does this problem need to be solved?

To work in the USA can be a challenging prospect for non immigrant workers. Most people take a path that includes pursuing a masters degree as it guarantees a stayback period of two years after starting a job during the opt period. The initial investments required for a masters degree are high. Many international students rely on education loans to pursue a masters degree and eventually have a debt to pay as soon as they graduate. Staying beyond the two year stayback mark can become crucial for many in terms of debt repayment. A job offer that has a higher probability of guaranteeing a H1B visa would be an important factor among others while accepting it. It is obvious that to gain experience in the USA beyond two years would make a strong overall career along with other advantages.

1.4 Dataset Information

This dataset contains five year's worth of H-1B petition data, with approximately 3 million records overall from the year 2011 to 2016. The data source is a Kaggle user's dataset https://www.kaggle.com/nsharan/h-1b-visa. The Office of Foreign Labor Certification (OFLC) generates program data, including data about H1B visas. The disclosure data is updated annually and is available on the website.

Features in the dataset

CASE STATUS : Status associated with the last significant event or decision. Valid values include Certified, Certified-Withdrawn, Denied, and Withdrawn.

EMPLOYER NAME : Name of employer submitting labour condition application.

SOC NAME : Occupational name associated with the SOC CODE. SOC CODE is the occupational code associated with the job being requested for temporary labour conditions, as classified by the Standard Occupational Classification (SOC) System.

JOB TITLE : Title of the job.

FULL TIME POSITION : Y = Full Time Position; N = Part Time Position.

PREVAILING WAGE : Prevailing Wage for the job being requested for temporary labour condition. The wage is listed at an annual scale in USD. The prevailing wage for a job position is defined as the average wage paid to similarly employed workers in the requested occupation in the area of intended employment. The prevailing wage is based on the employer's minimum requirements for the position.

YEAR : Year in which the H-1B visa petition was filed.

WORKSITE : City and State information of the foreign worker's intended area of employment.

LON: Longitude of the Worksite.
LAT: Latitude of the Worksite.

2. Problem Definition and Algorithms

2.1 Problem Definition

A dataset consisting of data related and relevant to the prediction of H1B visa is collected from the data source. It has detailed information about the employer who is recruiting and the information of individual employees who intend to work in the USA. The problem definition is to predict the probability of approval of the H1B visa for an employee depending on various factors relevant to both the employer as well as the employee. It is a binary classification problem. The output will be in the binary form predicting whether the visa has been approved or not along with its probability.

2.2 Algorithm Definitions

K Nearest Neighbor

K Nearest Neighbor (KNN) is a machine learning algorithm that is used for both classification and regression tasks. It is a type of instance-based learning, where the model makes predictions based on the similarities between the new data point and the training examples. Given a new data point, the algorithm finds the k nearest neighbours in the training data based on the distance metric, such as Euclidean or Manhattan distance. For classification, the algorithm assigns the class label of the majority of the k nearest neighbours to the new data point. The main advantage of the KNN algorithm is its simplicity and ease of implementation. It does not make any assumptions about the underlying distribution of the data and can handle non-linear decision boundaries. Moreover, KNN can be used for both classification and regression tasks and can be applied to a wide range of domains.

Decision Tree

A decision tree is a graphical representation of a decision-making process, which uses a tree-like model to illustrate a series of possible decisions and their outcomes. It is a tool used in various fields, such as business, finance, healthcare, and engineering, to help individuals or organisations make complex decisions by breaking them down into smaller, more manageable parts. Decision trees can become much more complex depending on the nature of the decision and the number of possible outcomes. The key advantage of using a decision tree is that it can help individuals or organisations make informed decisions by considering all the possible outcomes and their probabilities.

Random Forest

Random forest is a popular machine learning algorithm used for classification, regression, and other tasks in which data needs to be predicted or analysed. It is an ensemble learning method that combines multiple decision trees to create a more accurate and robust model. In Random Forest, multiple decision trees are constructed on different subsets of the training data using random selection of features and data samples. These trees are then combined to form a forest. When making a prediction, each tree in the forest votes on the class or value, and the final prediction is based on the majority vote. The algorithm can be trained using various techniques such as bagging and boosting. Overall, Random Forest is a powerful and versatile algorithm that has been successfully applied to a wide range of real-world problems.

Extreme Gradient Boosting

XGBoost (eXtreme Gradient Boosting) is a popular machine learning algorithm that belongs to the family of ensemble methods. It is used for both regression and classification problems and is known for its high accuracy and speed. XGBoost works by constructing multiple decision trees in a serial manner, where each subsequent tree learns from the errors made by the previous tree. The algorithm uses a gradient boosting technique to minimise the loss function and improve the performance of the model. XGBoost is a powerful and versatile algorithm that is widely used in industry and academia for solving complex machine learning problems.

3. Experimental Evaluation

3.1 Methodology

A machine learning project involves several iterative cycles of data preparation, analysis, model selection, training, and evaluation until the desired level of performance is achieved. It also requires collaboration between data scientists, domain experts, and stakeholders to ensure that the project meets the business requirements and objectives.

Import the required packages

numpy 1.24.2
import numpy as np

```
pandas  1.5.3
import pandas as pd

scikit-learn 1.2.1
import  scikit-learn

xgboost 1.7.4
import  xgboost as xgb

pickle  0.0.4
import pickle

streamlit 1.19.0
import streamlit as st
```

Load the raw data

```
df = pd.read_csv("h1b_2011_2016.csv")
df.head()
df = df.drop(['Unnamed: 0'], axis=1)
df.info()
```

Data Cleansing Process

The data has missing values in almost all of the columns. The number of missing values is not significant as compared to the overall data. The rows with missing values are dropped.

```
df.dropna(axis=0, inplace=True)
df.isna().sum()
```

The worksite column contains the state and the city the employee worked in. The column is split into two individual columns, state and city, and the original column is dropped.

```
df[['CITY', 'STATE']] = df['WORKSITE'].str.split(', ', 1, expand=True)
del df['WORKSITE']
```

Regex data preprocessing

```
df.rename(columns={'lon': 'LON', 'lat': 'LAT'}, inplace=True)
```

```
df['EMPLOYER_NAME'] = df['EMPLOYER_NAME'].str.replace("""[,./'")(*]""", "")
df['EMPLOYER_NAME'] = df['EMPLOYER_NAME'].str.replace("-", " ")

df['SOC_NAME'] = df['SOC_NAME'].str.replace("""[,./'")(*]""", "")
df['SOC_NAME'] = df['SOC_NAME'].str.replace("-", " ")

df['JOB_TITLE'] = df['JOB_TITLE'].str.replace("""[,./'")(*]""", "")
df['JOB_TITLE'] = df['JOB_TITLE'].str.replace("-", " ")

df[‘CASE_STATUS’] = df[‘CASE_STATUS’].str.upper()
df['EMPLOYER_NAME'] = df['EMPLOYER_NAME'].str.upper()
df['SOC_NAME'] = df['SOC_NAME'].str.upper()
df['JOB_TITLE'] = df['JOB_TITLE'].str.upper()
df['FULL_TIME_POSITION'] = df['FULL_TIME_POSITION'].str.upper()
df['CITY'] = df['CITY'].str.upper()
df['STATE'] = df['STATE'].str.upper()
```

The data contains multiple classes. Classes other than Certified class have lower values as compared to the entire data. To make it a binary classification problem, the Certified case status is converted into 1 and Certified-Withdrawn, Denied, Withdrawn, Rejected, Invalidated and Pending Quality and Compliance Review are assigned to 0 using label binarize. A new column ‘ACCEPT_REJECT’ is created having a binary class and the original column with multiple classes is dropped.

```
from sklearn.preprocessing import label_binarize
df['ACCEPT_REJECT'] = label_binarize(df['CASE_STATUS'], classes = ['CERTIFIED'])
del df['CASE_STATUS']
```

Convert the Full Time Position column data from categorical to numeric values using label encoding.

```
from sklearn.preprocessing import LabelEncoder
status_encoder = LabelEncoder()
df['FULL_TIME_POSITION']=status_encoder.fit_transform(df['FULL_TIME_POSITION'])
```

In order to get the data of various features that have a significant impact on the model, remove the data lower than the specified threshold for each feature. Depending on the number of applications made below the specified limit, the unique records from each feature are removed.

```
soc_values = df['SOC_NAME'].value_counts()
df = df[~df['SOC_NAME'].isin(soc_values[soc_values < 501].index)]
```

```
job_values = df['JOB_TITLE'].value_counts()
df = df[~df['JOB_TITLE'].isin(job_values[job_values < 1001].index)]

emp_values = df['EMPLOYER_NAME'].value_counts()
df = df[~df['EMPLOYER_NAME'].isin(emp_values[emp_values < 601].index)]

city_values = df['CITY'].value_counts()
df = df[~df['CITY'].isin(city_values[city_values < 601].index)]
```

As a result of the above step,, the records may drop down below the threshold limit. Such records are aggregated together and replaced by ‘Others’ in each column bringing them back above the specified threshold limit.

```
for x in soc_values[soc_values<500].index:
   df.loc[df['SOC_NAME']==x, 'SOC_NAME'] = 'OTHERS'

for x in job_values[job_values<500].index:
   df.loc[df['JOB_TITLE']==x, 'JOB_TITLE'] = 'OTHERS'

for x in emp_values[emp_values<600].index:
   df.loc[df['EMPLOYER_NAME']==x, 'EMPLOYER_NAME'] = 'OTHERS'

for x in city_values[city_values<700].index:
   df.loc[df['CITY']==x, 'CITY'] = 'OTHERS'

for x in state_values[state_values<10].index:
   df.loc[df['STATE']==x, 'STATE'] = 'OTHERS'
```

Flow Diagram

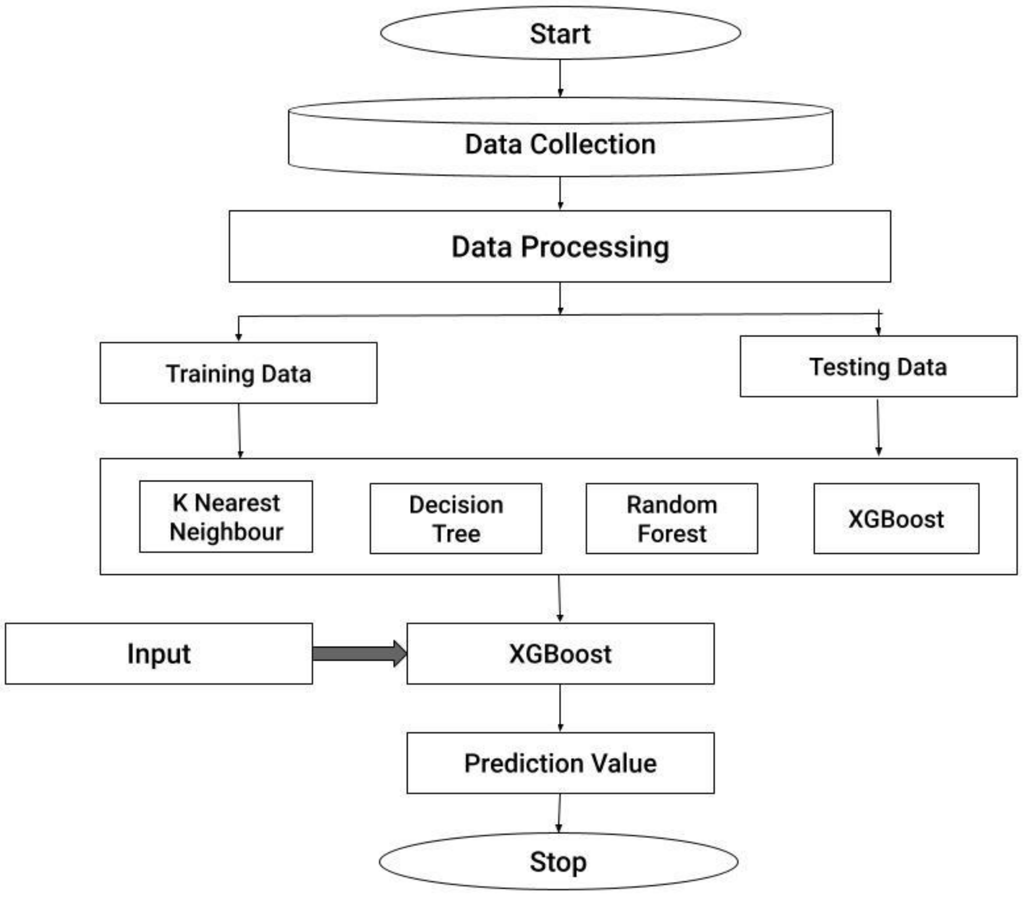


Fig 1 : Flow diagram of the machine learning project

3.2 Exploratory Data Analysis

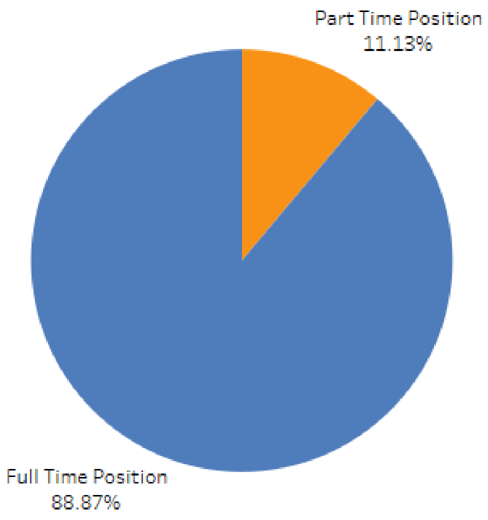


Fig 2 : Pie Chart of Positions of Employees

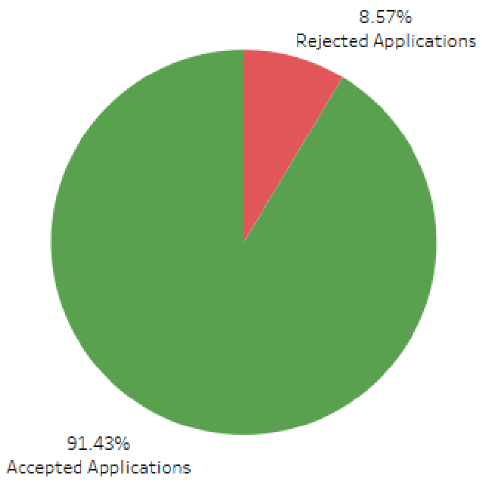


Fig 3 : Pie Chart of Accepted/Rejected Applications

The percentage of full time and part time positions of the employees whose applications have been petitioned for the H1B visa are represented using a pie chart in figure 2. The amount of part time positions is less than around 10% that of full time positions. This is because the employers prefer to sponsor visas to those employees who are recognized as an asset and value to their company. This requires full time dedication of the employees along with other factors.

Figure 3 is a pie chart representing the percent accepted and percent rejected applications out of the total applications made during the five years from 2011 to 2016. It shows the high imbalance between the binary classes of the dataset. The number of rejected applications is less than 10% of the number of accepted applications. To balance this data, implementation of SMOTE technique is necessary.

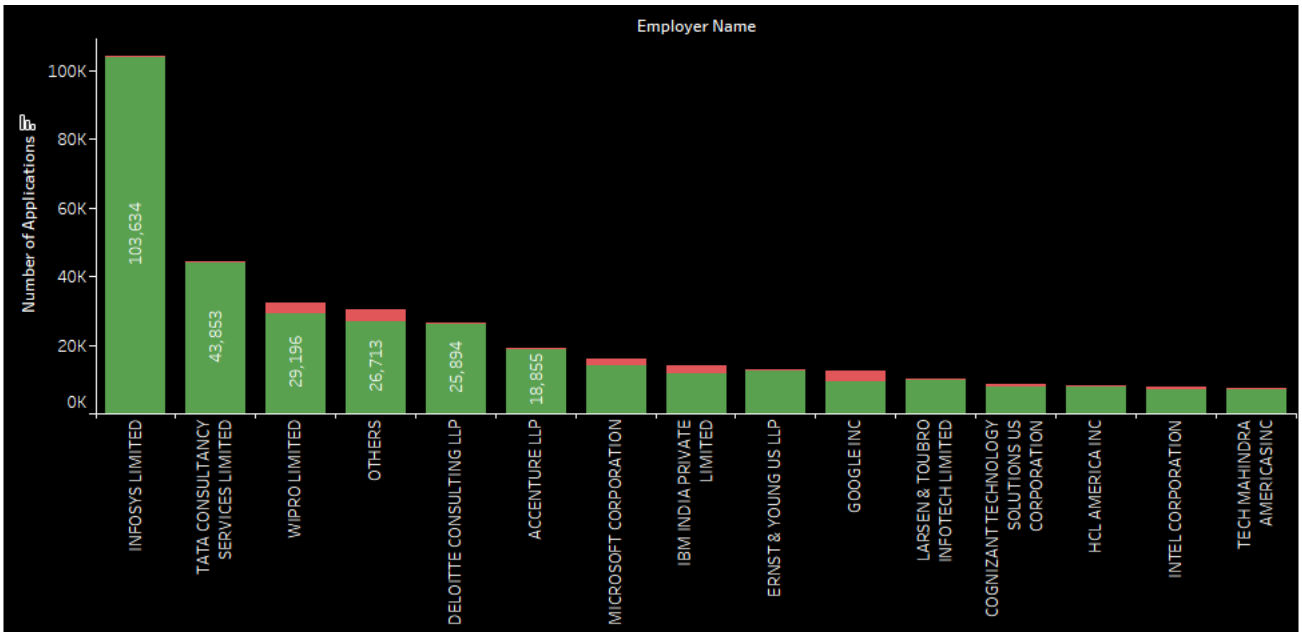


Fig 4 : Bar Graph of Top 15 Companies by Number of Applications

Top 15 companies that petition for the most number of applications in the entire USA are represented in the figure 4 vertical bar graph. The number of accepted and rejected applications is displayed. Indian conglomerates having offices in the USA are the top applicants. They are followed by the 'MAANG' companies representing the IT sector and the 'Big Four' from the Finance sector.

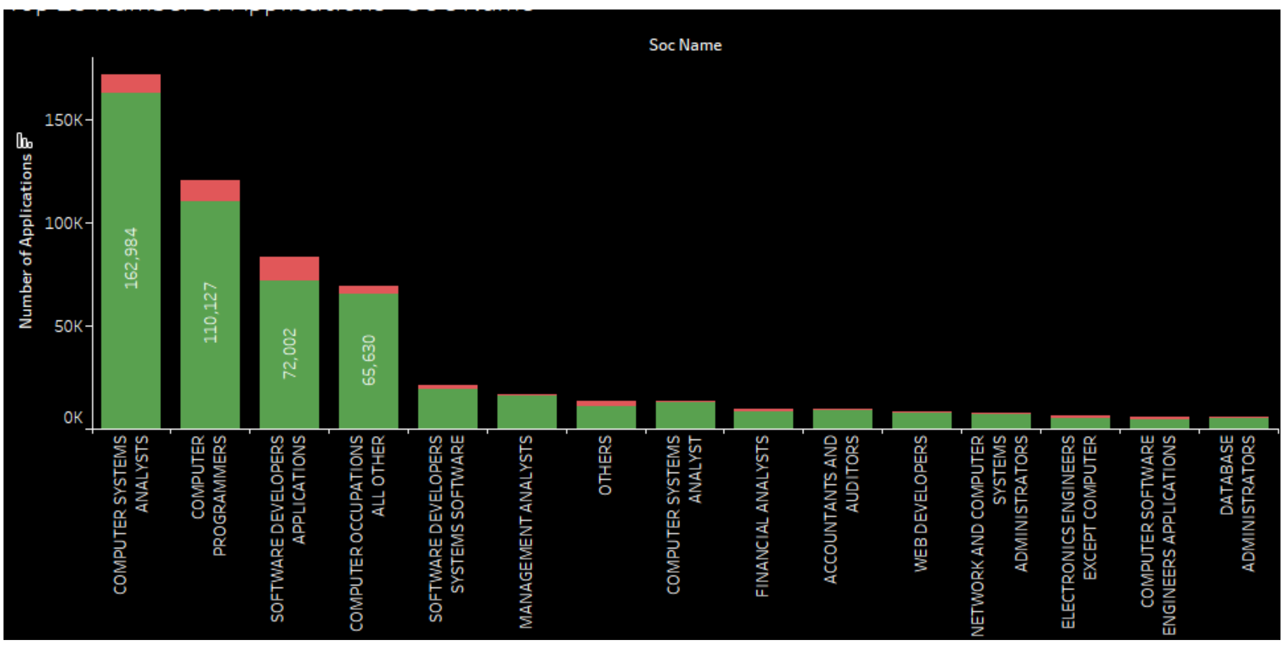


Fig 5 : Bar Graph of Top 15 SOC by Number of Applications

The 2018 Standard Occupational Classification (SOC) system is a federal statistical standard used by federal agencies to classify workers into occupational categories for the purpose of collecting, calculating, or disseminating data. Detailed occupations in the SOC with similar job duties, and in some cases skills, education, and/or training, are grouped together. Employees working in a technical field from the IT sector such as software developers, programmers and analysts dominate over other SOC groups.

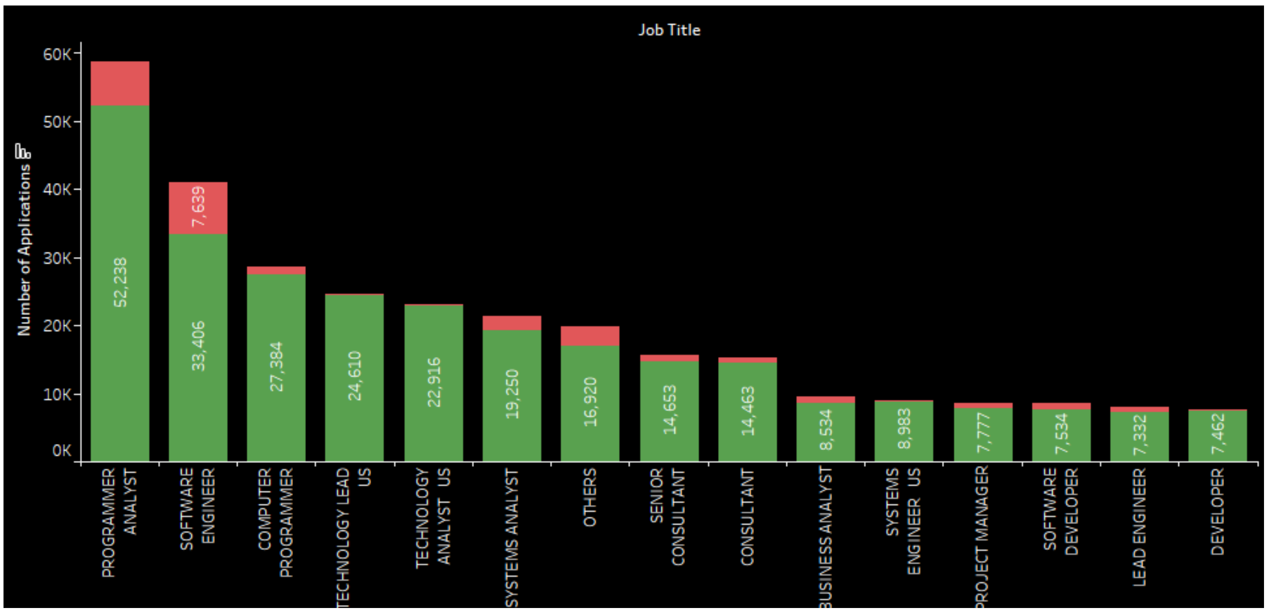


Fig 6 : Bar Graph of Top 15 Job Titles by Number of Applications

Figure 6 shows a vertical bar graph of top 15 job profiles that get their applications petitioned for the H1B visa process. This in turn means that the employees having a job profile of an IT technician are most likely to get a sponsorship from their employer. The developers, programmers and analysts dominate, naturally, because of the huge IT hub all over the USA.

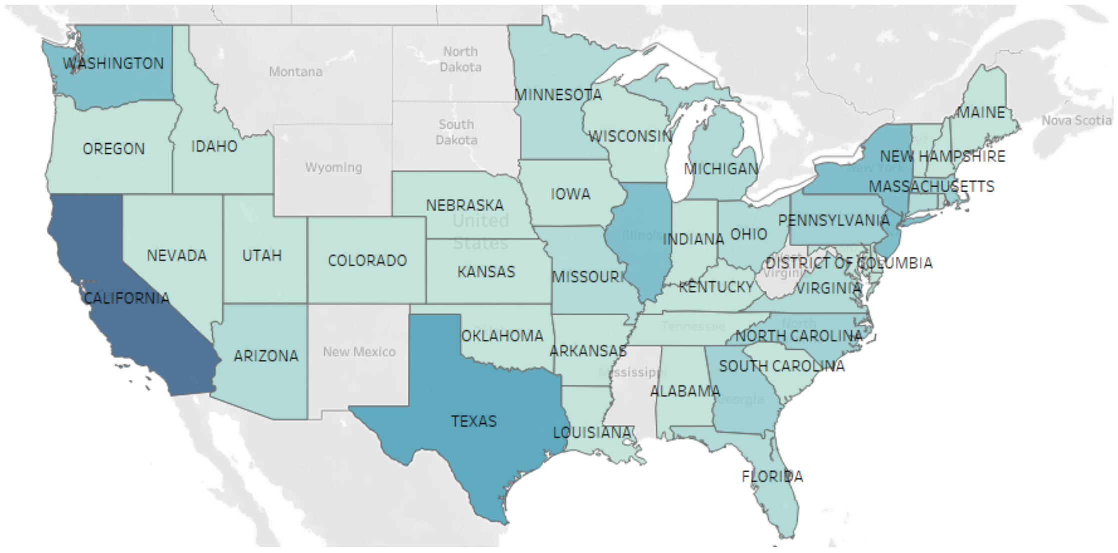


Fig 7 : Map of Number of Applications by State

A map of the USA highlights the state wise number of applications petitioned by the companies in figure 7. The state of California makes the most applications. This can be a result of many companies having their headquarters established in California. Many Indian companies who lead in the number of applications petitioned as well as the US conglomerates or the 'MAANG' companies have offices in this state. Other states such as Texas, Washington, New York, North Carolina that have huge IT hubs lead over other states.

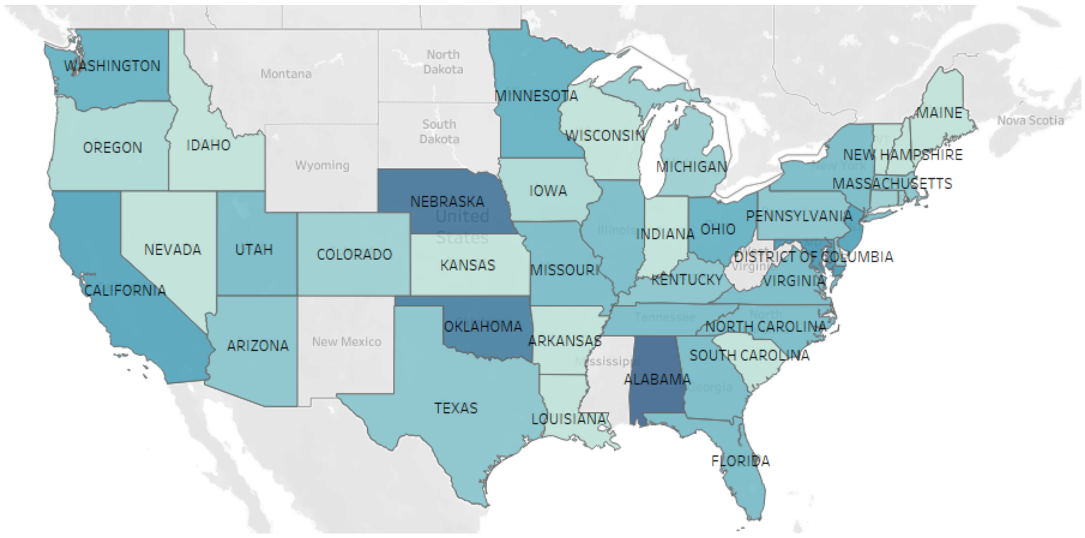


Fig 8 : Map of Average Salary by State

Figure 8 shows the map of the USA with the state wise aggregation of the average salary. The states of Nebraska, Oklahoma and Alabama have the highest average salary. Though these states do not petition as many applications, the companies seem to offer higher salaries resulting in higher average salaries than other states. California is the next state with higher average salary, a result of its huge demand for technical employees having higher salaries than other job professionals.

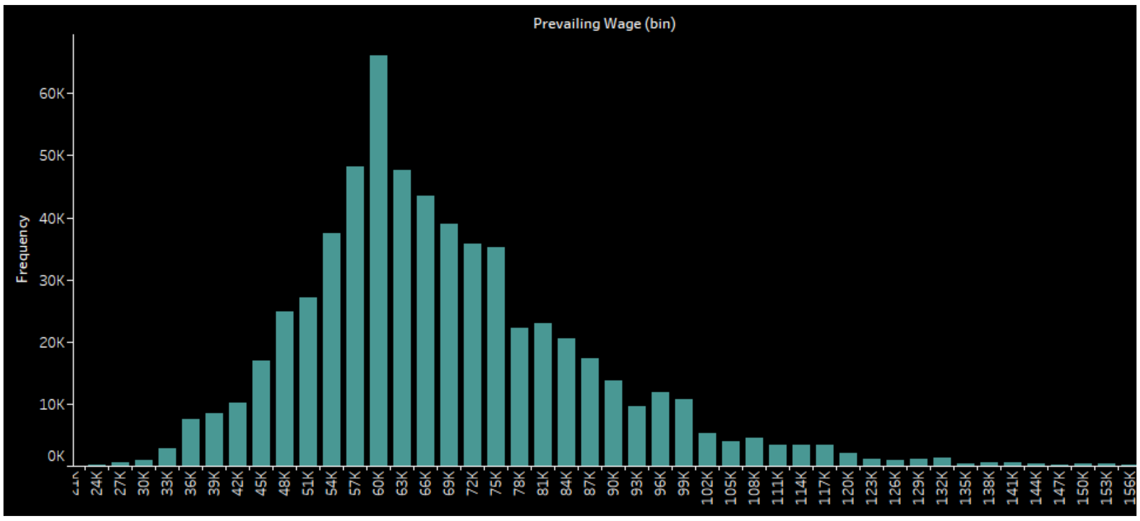


Fig 9 : Salary Distribution

Figure 9 is an histogram that represents the salary distribution of the international employees working in the USA. The annual salary peaks at around 60k. After 60k, as the salary increases, the frequency or the amount of applications with higher salaries start to decrease. Therefore, it is a right skewed distribution.

4. Model Creation and Evaluation

4.1 Data Preprocessing

The categorical features are changed into numeric values using categorical codes. All the features from the dataset are converted to category type. Then each unique record is mapped to a numeric code that can be used for the machine learning models.

```
df['EMPLOYER_NAME'] = df['EMPLOYER_NAME'].astype("category")
df['SOC_NAME'] = df['SOC_NAME'].astype("category")
df['JOB_TITLE'] = df['JOB_TITLE'].astype("category")
df['STATE'] = df['STATE'].astype("category")
df['CITY'] = df['CITY'].astype("category")
```

```
df['EMPLOYER_NAME'] = df['EMPLOYER_NAME'].cat.codes
df['SOC_NAME'] = df['SOC_NAME'].cat.codes
df['JOB_TITLE'] = df['JOB_TITLE'].cat.codes
df['STATE'] = df['STATE'].cat.codes
df['CITY'] = df['CITY'].cat.codes
```

Feature Selection

The covariance and correlation between all the features is checked

```
df.cov()
df.corr()
```

The multicollinearity of each column depending on other features in the dataset is checked using variance inflation factor.

```
from statsmodels.stats.outliers_influence import variance_inflation_factordef
calc_vif(X):
   vif = pd.DataFrame()
   vif["variables"] = X.columns
   vif["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
   return(vif)

X = df.iloc[:,:-1]
calc_vif(X)
```

The best suited columns from the dataset are checked using SelectKBest.

```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2, f_regression
select = SelectKBest(score_func=chi2, k=6)
z = select.fit_transform(x,y)
filter = select.get_support()
features = x.columns

print("All features:")
print(features)
print("Selected best 6:")
print(features[filter])
```

Feature selection depending on the above processes results in the selection of various independent variables from the entire dataset.

Independent variables = x
Dependent variable = y

```
x = df.drop(['FULL_TIME_POSITION', 'YEAR', 'LON', 'LAT', 'ACCEPT_REJECT'], axis=1)
y = df['ACCEPT_REJECT']
```

Split the data

Split the data into train and test datasets. The whole dataset is split using 8:2 ratio for train and test respectively.

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=1)
```

The dataset is highly imbalanced. The records having target variable value as 0 is less than 10 percent of the records with 1 as the target variable. Therefore, Synthetic Minority Over-sampling Technique is used to handle the class imbalance. SMOTE is a technique that generates synthetic samples of the minority class by creating new examples that are similar to existing minority class examples. SMOTE works by randomly selecting a minority class example and finding its k nearest neighbours. It then generates new synthetic examples by interpolating between the original example and its nearest neighbours. This process is repeated until the desired level of over-sampling is achieved.

```
from imblearn.over_sampling import SMOTE
sm = SMOTE()
x_train_sm, y_train_sm = sm.fit_resample(x_train, y_train.ravel())
```

4.2 Model Creation

K Nearest Neighbour, Decision Tree, Random Forest and Extreme Gradient Boost models were created and evaluated. Out of these, the Extreme Gradient Boost model is chosen and implemented. Even though its accuracy is less than that of the Decision Tree and Random Forest models by 2-3%, it has a higher precision, recall and f1-score values for both the binary classes. This is an important factor considering our dataset is imbalanced and handled using SMOTE. Moreover, the XGBoost model has the highest roc score of any model.

```
def create_model_xgb():
    import xgboost as xgb
    model = xgb.XGBClassifier()
    model.fit(x_train_sm, y_train_sm)
    return model

model_xgb = create_model_xgb()
```

4.3 Model Evaluation

Model evaluation is the process of assessing the performance of a machine learning model that predicts categorical outcomes. The evaluation of a classification model is important to determine whether the model is accurate and reliable in making predictions on new, unseen data. Here are some common evaluation metrics for classification models -

1. Confusion matrix: A table that compares the actual and predicted classifications of the model. It shows the number of true positives, false positives, true negatives, and false negatives.
2. Accuracy: The proportion of correct predictions over the total number of predictions. It is calculated as (TP + TN) / (TP + TN + FP + FN).
3. Precision: The proportion of true positives over the total number of positive predictions. It is calculated as TP / (TP + FP).
4. Recall (also known as sensitivity): The proportion of true positives over the total number of actual positives. It is calculated as TP / (TP + FN).
5. F1 score: The harmonic mean of precision and recall. It is calculated as 2 * (precision * recall) / (precision + recall).
6. Area Under the Receiver Operating Characteristic Curve (AUC-ROC): A metric that measures the model's ability to distinguish between positive and negative classes. It plots the true positive rate against the false positive rate and calculates the area under the curve.

```
from sklearn.metrics import confusion_matrix, classification_report
def evaluate_model(model, model_name):
    print(f"--- evaluation report of {model_name} ---")
    y_pred = model.predict(x_test)

    print(confusion_matrix(y_test, y_pred))
    print(classification_report(y_test, y_pred))

from sklearn.metrics import roc_curve, roc_auc_score
def roc_evaluation(model, model_name):
    print(f"--- roc evaluation report of {model_name} ---")

    y_pred_probabilities = model.predict_proba(x_test)[:, 1]
    fpr, tpr, thresholds = roc_curve(y_test, y_pred_probabilities)

    plt.plot(fpr, tpr, label="RoC Curve")
    plt.legend()

    roc_score = roc_auc_score(y_test, y_pred_probabilities)
    print(f"roc score = {roc_score}")

evaluate_model(model_xgb, "XGBoost")
roc_evaluation(model_xgb, "XGBoost")
```

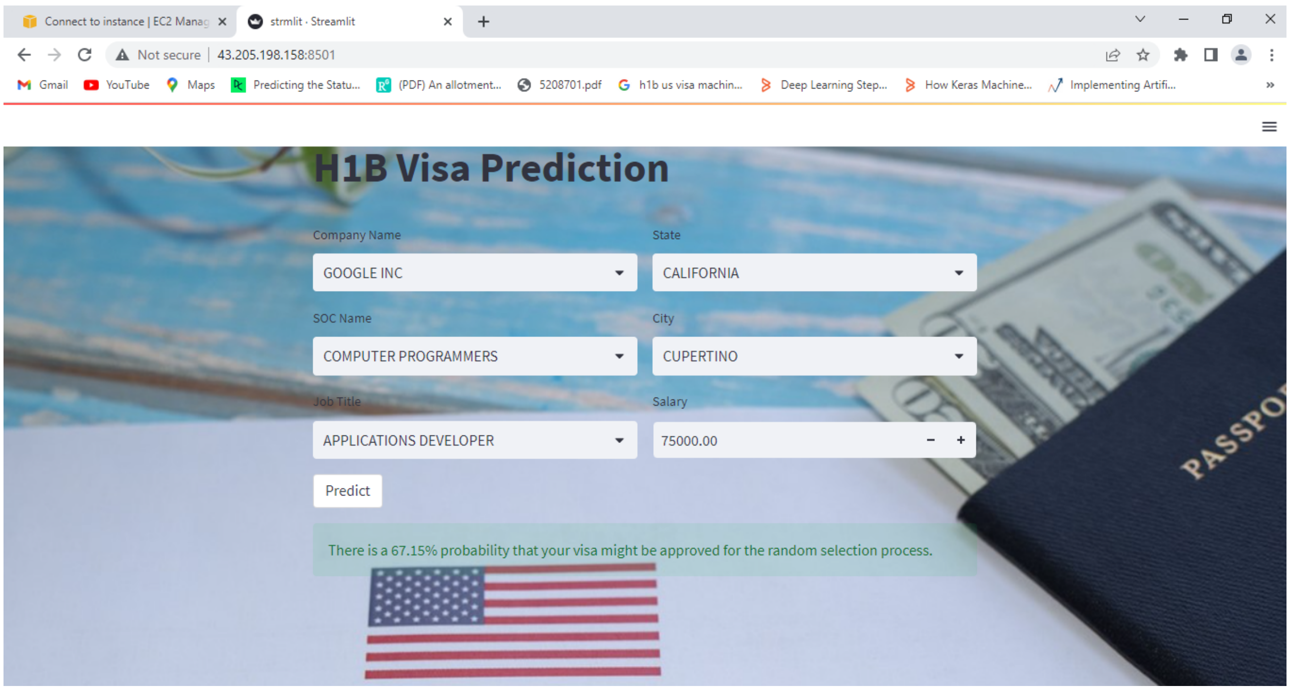
4.4 Saving the model

The pickle module in Python provides functions for pickling (serialising) and unpickling (deserializing) Python objects. The pickled representation of an object can be stored in a file or transferred over a network, and then later unpickled to recreate the original object.

```
import pickle
model_file = open('h1b_prediction_model_rf2.pk', 'wb')
pickle.dump(model_rf, model_file)
model_file.close()
```

5. Graphical User Interface

Streamlit is an open-source Python library that allows developers to create interactive web applications for machine learning and data science projects. It provides a simple and intuitive interface that allows developers to create interactive and customizable web-based user interfaces. With Streamlit, developers can easily create a variety of visualisations, such as charts, graphs, tables, and maps, and incorporate machine learning models and data analysis into the web application. Streamlit also provides built-in components, such as sliders, drop-down menus, and buttons, which can be integrated into the application. Streamlit also provides easy deployment options, making it simple to share the web application with others. It supports deploying web applications to a variety of platforms, including the cloud and local servers.



H1B Visa Prediction

Company Name: GOOGLE INC	State: CALIFORNIA
SOC Name: COMPUTER PROGRAMMERS	City: CUPERTINO
Job Title: APPLICATIONS DEVELOPER	Salary: 75000.00

There is a 67.15% probability that your visa might be approved for the random selection process.

6. Future Work and Conclusion

6.1 Future Work

The H1b visa process is a two step process wherein the application is first petitioned as the Labor Condition Application (LCA) with the U.S. Department of Labor. The current machine learning model predicts the probability of the outcome of this LCA. The applications approved by the U.S. Department of labour then are filed with the U.S. Citizenship and Immigration Services (USCIS). The applications approved by the USCIS are considered to be on a random basis without the elements of bias for any features or factors. A future project on data analysis on this ‘lottery system’ may give a better insight into whether that is the actual case. A bias, if exists, can be determined. A machine learning model can be implemented to exploit the bias and predict the status of an applicant’s visa outcome.

6.2 Conclusion

- The state of California leads the way in filing as well as granting the petitions for the H1B non immigrant work visas in the USA.
- Major companies that employ international employees are from the IT and Finance sectors.
- Indian conglomerates lead the way in filing petitions for the H1B visas followed by US conglomerates.

- Majority of the applicants have a job profile in the IT sector. The companies mostly hire and sponsor those international employees who are skilled in the technical field. This improves the chance of getting a visa, as 90% of the visas are approved for the next step of visa procedure.
- The salary distribution peaks at 60k for the international employees. An annual salary between 55k to 75k is most likely what a H1B visa petitioner will earn in the USA.

Overall, an non immigrant employee working in any state having an IT hub for a US conglomerate or that of Indian origin as an IT professional, earning an annual salary of around 70k has got a high probability of visa being approved for the random selection process.