VISA APPROVAL PREDICTION

1.INTRODUCTION:-

1.1.Overview:-

Over 2 Million visa petitions are filed by the employers each year and only 65000 petitions are approved. So, the goal is to explore the petitions filed and their outcomes for the past six years i.e., from 2011 to 2016, and to find a pattern to predict the outcome by using a predictive model developed using Machine Learning techniques. In the Guided Project, our goal is to predict the outcome of H-1B visa applications that are filed by many professional foreign nationals every year.

1.2.Purpose:-

H-1B is a type of non-immigrant visa in the United States that allows foreign nationals to work in occupations that require specialized knowledge and a bachelor's degree or higher in the specific specialty. This visa requires the applicant to have a job offer from an employer in the US before they can file an application to the US immigration service (USCIS). We believe that this prediction algorithm could be a useful resource both for the future H-1B visa applicants and the employers who are considering sponsoring them.

2.LITERATURE SURVEY:-

2.1. Existing problem:-

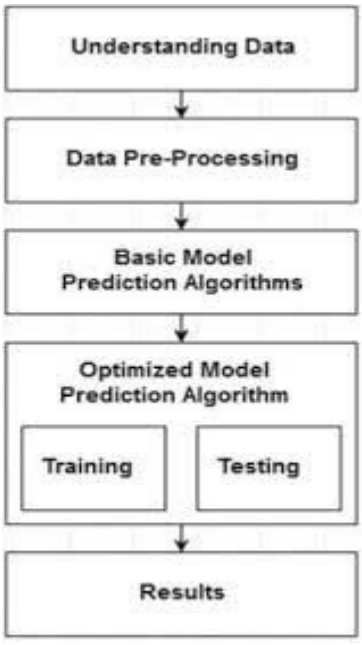
An H-1B visa or status is often denied or refused because the petitioner—that is, the employer sponsoring the H-1B visa—does not appear to be a real. In past years, this became a particular problem in cases where the H-1B petition stated that the employee would work offsite at a client location. USCIS wanted to know whether the employee would truly be working for the petitioning employer, or whether the employer was trying to get around the rules by acting as a "job shop," placing employees on subcontracting assignments.

2.2. Proposed solution:-

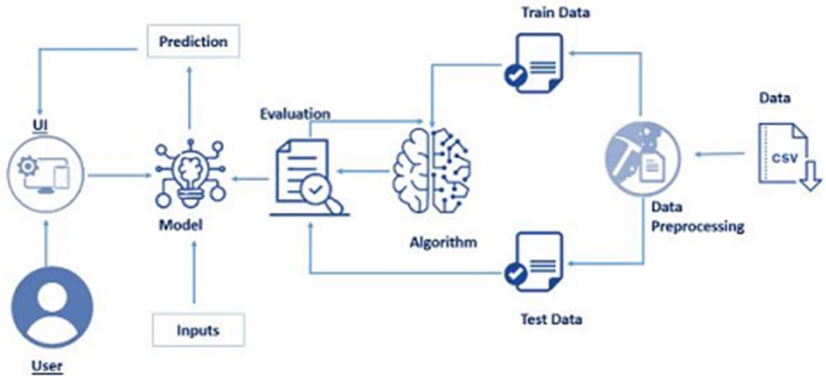
We predict the visa approval by application position and their wage and occupation code for the employment and we predict whether the application denied, certified-withdrawn, withdrawn, pending quality and compliance review, invalid or rejected.

3.THEORITICAL ANALYSIS:-

3.1.Block diagram:-



3.2.Technical Architecture:



3.3.Hardware/software designing

Software Requirements:

- OS – Windows XP,7,8,10
- Jupyter Software
- Spyder Software
- Anaconda Command Prompt

Hardware Components:

- Processor – i3
- Hard Disk Storage – 10 GB
- RAM – 1GB

4.EXPERIMENTAL INVESTIGATIONS:-

Some denials asserted that using an entry-level prevailing wage (Level 1 in the Department of Labor's system), which is

appropriate for many jobs that require a bachelor's degree and up to two years of experience, means that the job cannot possibly require a bachelor's degree. That was the focus in 2017, but appears as of 2020 no longer to be an automatic ground to question whether the job can qualify for an H-1B petition.

In 2018, USCIS began denying H-1B petitions for Computer Systems Analyst, Market Research Analyst, and Financial Analyst jobs, which previously had qualified as H-1B specialty occupations. Setting aside the illogical reasoning of these decisions, employers need to be prepared for yet more surprising reasons for H-1B denials under the current administration. And, employers continue to sue USCIS over arbitrary denials of H-1B petitions.

5.Results:-

The result is we predict whether the application is denied, rejected, invalid, certified or pending.

6.ADVANTAGES AND DISADVANTAGES:-

Advantages:-

- We are going to predict whether the visa will be approved or not.
- Money will be saved.
- We will know approval of visa before they announce.
- With the help visa we can travel other countries.
- Person will get relived from tension.
- Lot of time will be saved.

Disadvantages:

- Prediction of visa approval may not be true.
- It will not predict the future predictions are based on past data.
- Money will be wasted.
- Person may not able travel, to the other countries.

7.APPLICATIONS:-

- PREVAILING_WAGE: This is the amount of prevailing wage for the position that is petitioned for in this labor condition application..
- YEAR: The fiscal year in which this labor condition application was filed.
- FULL_TIME_POSITION: This field states whether the position being petitioned for is a full time one or not.

8.CONCLUSION :-

Finally,after giving details and by clicking predict it shows the result whather the application is denied,pending or certified.

9.FUTURE SCOPE :-

Supplemental data concerning the Standard Occupational Classification (SOC) can be gathered and used in coordination with this data set to obtain a more comprehensive analysis of how the H-1B Visa selection process works. By using the wage evaluations and ranges under SOC, the wage attribute in this data set can be correctly put in to a range of salaries which can then be used to

classify the visa petitions based on occupation roles rather than location wise. In addition, other classification algorithms other than the discriminative models can be experimented with this testbed and their performances can also be analyzed.

10.BIBILOGRAPHY:-

1. H-1B Visa Petitions 2011-2016 — Kaggle, Oct 2017https://www.kaggle.com/nsharan/h-1b-visa/data.

2. *H-1B Visa Data Analysis and Prediction by using K-means Clustering and Decision Tree Algorithms*, [online] Available: https://github.com/Jinglin-LI/H1B-VisaPrediction-by-Machine-LearningAlgorithm/blob/master/H1B\%20Prediction\%20Research\%20Report.pdf.

3. F. Pedregosa, et. al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, 2825-2830.

Source Code(Model Building):

```
import sys

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from collections import Counter as c

from sklearn.preprocessing import LabelEncoder

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import
accuracy_score,classification_report,confusion_matrix
import pickle
data = pd.read_csv('h1b_kaggle.csv')
data.shape
data
data.head()
data.tail()
data.info()
data.CASE_STATUS.value_counts()
plt.figure(figsize=(15,7))
data.CASE_STATUS.value_counts().plot(kind='bar')
plt.title("Number of Applications")
plt.show()
plt.figure(figsize=(12,6))
sns.set(style="whitegrid")
sns.countplot(x='FULL_TIME_POSITION',data=data)
plt.title('Number of applications made for Full Time Position')
plt.ylabel('Number of Partition made')
```

```
plt.show()

top_emp = list(data['EMPLOYER_NAME'][data['YEAR'] >=
2015].groupby(data['EMPLOYER_NAME']).count().sort_values(as
cending=False).head(10).index)


byempyear = data[['EMPLOYER_NAME', 'YEAR',
'PREVAILING_WAGE']][data['EMPLOYER_NAME'].isin(top_emp)
]


byempyear = byempyear.groupby([data['EMPLOYER_NAME'],
data['YEAR']])

plt.figure(figsize=(12,7))


markers=['o','v','^','<','>','d','s','p','*','h','x','D','o','v','^','<','>','d',

        's','p','*','h','x','D']


for company in top_emp:

    tmp = byempyear.count().loc[company]

    plt.plot(tmp.index.values, tmp["PREVAILING_WAGE"].values,
label=company,

            linewidth=2,marker=markers[top_emp.index(company)])

plt.xlabel("Year")

plt.ylabel("Number of Applications")

plt.legend()

plt.title('Number of Applications of Top 10 Applicants')
```

```
plt.show()

data.WORKSITE.value_counts()

data=data[data['PREVAILING_WAGE']<=500000]

by_emp_year=data[['EMPLOYER_NAME',"YEAR",'PREVAILING
_WAGE']][data['EMPLOYER_NAME'].isin(top_emp)]

by_emp_year=by_emp_year.groupby([data['EMPLOYER_NAME']
,data["YEAR"]])

data.isnull().any()

data.isnull().sum()

data['SOC_NAME']=data['SOC_NAME'].fillna(data['SOC_NAME'].
mode()[0])

data['EMPLOYER_NAME']=data['EMPLOYER_NAME'].fillna(data[
'EMPLOYER_NAME'].mode()[0])

data['JOB_TITLE']=data['JOB_TITLE'].fillna(data['JOB_TITLE'].mo
de()[0])

data['lon']=data['lon'].fillna(data['lon'].mean())

data['lat']=data['lat'].fillna(data['lat'].mean())

data.isnull().sum()

data.CASE_STATUS.value_counts()

data['CASE_STATUS']=data['CASE_STATUS'].map({'CERTIFIED
' : 0 , 'CERTIFIED-WITHDRAWN' : 1,'DENIED' : 2,'WITHDRAWN'
: 3,'PENDING QUALITY AND COMPLIANCE REVIEW -
UNASSIGNED' : 4, 'REJECTED':5,'INVALIDATED' : 6})

data['CASE_STATUS']

data.CASE_STATUS.value_counts()
```

```
data.isnull().any()

data['FULL_TIME_POSITION']=data['FULL_TIME_POSITION'].map({'N' : 0,'Y' : 1})

data['FULL_TIME_POSITION']

data.head()

data['SOC_NAME1']='others'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('computer','software')]='it'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('chief','mangement')]='manager'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('mechanical')]='mechanical'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('database')]='database'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('sales','market')]='scm'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('financial')]='finance'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('public','fundraising')]='pr'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('education','law')]='administrative'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('auditors','compliance')]='audit'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('distribution','logistics')]='scm'
```

```
data['SOC_NAME1'][data['SOC_NAME'].str.contains('recruiters','human')]='hr'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('agricultural','farm')]='agri'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('construction','architectural')]='estate'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('forencsic','health')]='medical'

data['SOC_NAME1'][data['SOC_NAME'].str.contains('teachers')]='education'

data=data.drop(['Unnamed: 0','EMPLOYER_NAME','SOC_NAME','JOB_TITLE','WORKSITE','lon','lat'],axis=1)

data

data.isnull().any()

le=LabelEncoder()

data['SOC_N']=le.fit_transform(data['SOC_NAME1'])

data=data.drop(['SOC_NAME1'],axis=1)

data

sns.heatmap(data.corr(),annot=True,cmap='RdYlGn',annot_kws={"size":15})

plt.title('Correlation')

plt.show()

x=data.iloc[:,1:5].values

y=data.iloc[:,0].values
```

```
x

y

x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.75,ra
ndom_state=1)

x_train,x_test

y_train,y_test

lr=LogisticRegression()

lr.fit(x_train,y_train)

y_pred_lr=lr.predict(x_test)

y_pred_lr

c(y_pred_lr)

y_test

print(accuracy_score(y_test,y_pred_lr))

print(confusion_matrix(y_test,y_pred_lr))

print(classification_report(y_test,y_pred_lr))

rf=RandomForestClassifier()

rf.fit(x_train,y_train)

y_pred_rf=rf.predict(x_test)

y_pred_rf

c(y_pred_rf)

y_test

print(accuracy_score(y_test,y_pred_rf))

print(confusion_matrix(y_test,y_pred_rf))
```

```
print(classification_report(y_test,y_pred_rf))
dt=DecisionTreeClassifier()
dt.fit(x_train,y_train)
y_pred_dt=dt.predict(x_test)
y_pred_dt
c(y_pred_dt)
y_test
print(accuracy_score(y_test,y_pred_dt))
print(confusion_matrix(y_test,y_pred_dt))
print(classification_report(y_test,y_pred_dt))
pickle.dump(dt,open('visarf.pkl','wb'))  ###
knn=KNeighborsClassifier(n_neighbors=4)
knn.fit(x_train,y_train)
y_pred_knn=knn.predict(x_test)
c(y_pred_knn)
y_test
print(accuracy_score(y_test,y_pred_knn))
print(confusion_matrix(y_test,y_pred_knn))
print(classification_report(y_test,y_pred_knn))
```

Source Code(Application Building):

```
import numpy as np
import pandas as pd
from flask import Flask, request, render_template
```

```
import pickle
import os

app = Flask(__name__)
model = pickle.load(open('Visarf.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('Visa_Approval.html')

@app.route('/predict',methods=['POST'])
def predict():
    input_features = [float(x) for x in request.form.values()]
    features_value = [np.array(input_features)]

    features_name = ['FULL_TIME_POSITION', 'PREVAILING_WAGE',
'YEAR','SOC_N']

    df = pd.DataFrame(features_value, columns=features_name)
    output = model.predict(df)
    #output=np.argmax(output)
    print(output)


    return render_template('resultVA.html', prediction_text=output)

if __name__ == '__main__':

    app.run(debug=True)
```

Output:

Anaconda Prompt (anaconda3) - python app.py

```
(base) C:\Users\DELL>cd C:\Users\DELL\Visa_Approval_Prediction

(base) C:\Users\DELL\Visa_Approval_Prediction>python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with windowsapi reloader
 * Debugger is active!
 * Debugger PIN: 638-746-385
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```


H-1B VISA PREDICTION
YES(FULL-TIME)
50000
2017
Education
Predict


H-1B Visa Predictor
A Machine Learning Web App Built with Flask
Prediction: 'CERTIFIED'