Faddy Sunna
ID# 100142187

Phys 243 Midterm

May 22, 2019

Climate Model Simulation Crashes Data Set

1. Data processing

The data had to be cleaned and separated into columns at first. Microsoft Excel was used to pre-process the data. The first two columns (study ID, and simulation ID) were removed because they are not relevant to the outcome. The goal is to predict climate model simulation outcomes (column 19, fail or succeed) given scaled values of climate model input parameters (columns 1-18). Columns 1-18: values of 18 climate model parameters scaled in the interval [0, 1] Column 19: simulation outcome (0 = failure, 1 = success)

2. Training and testing datasets

Sklearn train_test_split was used to separate the data into train and test datasets. The test dataset is 15% of the training set. A randomly chosen training and testing dataset is generated each time the program is run.

3. Model type: KNN Classification

This dataset will need to be approached as a Classification problem, with a boolean output. KNN was used to predict the outcome (0 = fail, 1 = succeed), depending on the 18 independent variables.

4. Model Evaluation

Choosing the optimal K neighbors:
A for loop was used to iterate through K values, and measure the accuracy of each. Accuracy was then converted to misclassification error. The K (optimal_k) value with the lowest misclassification error is then passed into the KneighborsClassifier function.

The KneighborsClassifier function is then run 10,000 times, and the accuracy is averaged out. Which was 92.8% accuracy.

The ExtraTreesClassifier function was used to measure the importance of each variable on the outcome. The two most important variable with an importance of 0.1368148 and 0.13702292, where: vconst_corr and vconst_2, all other variables had a an importance of < .1.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	vconst_corr	vconst_2	vconst_3	vconst_4	vconst_5	vconst_7	ah_corr	ah_bolus	slm_corr	efficiency_factor	tidal_mix_max	vertical_decay_scale	convect_corr	bckgrnd_vdc1	bckgrnd_vdc_ban	bckgrnd_vdc_eq	bckgrnd_vdc_psim	Prandtl	outcome
2	0.859036206	0.927824536	0.252865622	0.298838311	0.1705213	0.735936041	0.428325428	0.567946942	0.4743696	0.245674855	0.104225865	0.869090703	0.997518496	0.448620077	0.307521787	0.858310365	0.79699724	0.869893038	0
3	0.606041025	0.457728363	0.359448423	0.306957377	0.843330767	0.934850661	0.444572488	0.828014925	0.296617753	0.616869902	0.975785581	0.914343667	0.845247142	0.864151868	0.346712689	0.356573417	0.438447189	0.512256144	1
4	0.997599777	0.373238487	0.517399356	0.504992546	0.618903336	0.605570823	0.74622533	0.195928292	0.815666935	0.679355028	0.803413076	0.643995161	0.718441133	0.924775074	0.315371406	0.250642371	0.285635527	0.365857964	1
5	0.783407859	0.104055314	0.197532695	0.421837159	0.742055668	0.490827882	0.005525437	0.392123267	0.010014895	0.4714627	0.597878903	0.761658752	0.362750561	0.912819094	0.977971175	0.845921227	0.699430932	0.475986735	1
6	0.406249531	0.513199297	0.061811577	0.635836719	0.844797517	0.441502227	0.191926451	0.487546116	0.358533581	0.551543235	0.743876519	0.312349434	0.650222833	0.522261016	0.043544765	0.376660111	0.280097759	0.132282876	1
7	0.04137947	0.629025938	0.303380105	0.813407572	0.222817123	0.971206033	0.60977829	0.647803859	0.737913873	0.440943207	0.035982366	0.615867667	0.017487184	0.932319515	0.329318038	0.954122776	0.135378915	0.294804796	1
8	0.161050068	0.548838498	0.153583456	0.654415397	0.140346213	0.796645887	0.405839624	0.662635625	0.049426685	0.578519494	0.264854683	0.959217373	0.698106832	0.467358949	0.637077696	0.011251331	0.147325157	0.213814247	1
9	0.415298986	0.898731363	0.931821763	0.916647939	0.399105638	0.009445115	0.84625746	0.68377254	0.397306457	0.886768087	0.522427847	0.694773565	0.886522318	0.411672883	0.481108054	0.92654637	0.02643087	0.092739552	1
10	0.166757878	0.352971856	0.988120611	0.287070317	0.5636256	0.402707875	0.380931733	0.479191238	0.060165294	0.236524445	0.290483708	0.391833072	0.254944302	0.488399586	0.053683565	0.862225604	0.415055139	0.487125899	1
11	0.655626079	0.413930717	0.805288478	0.163485864	0.861901695	0.947594532	0.546563676	0.426141006	0.417080309	0.945609574	0.325387509	0.666526808	0.374270055	0.100291351	0.21329028	0.222859671	0.007286213	0.420027362	1
12	0.590007977	0.293692412	0.423497844	0.329803463	0.4578377	0.829782296	0.497766735	0.159440334	0.971121865	0.492342538	0.084268972	0.974333948	0.926423909	0.295425919	0.804212452	0.870840009	0.546295023	0.884871079	1
13	0.881929079	0.424916157	0.903222367	0.173305914	0.791005372	0.476183524	0.681175841	0.905843852	0.828415466	0.083717329	0.520566651	0.072915346	0.948094685	0.999615891	0.728459115	0.285888356	0.210889643	0.833589868	1
14	0.961010262	0.976906806	0.857928773	0.614971227	0.615515929	0.352769204	0.833960006	0.095091749	0.230892542	0.954833278	0.577181749	0.78351143	0.530424645	0.175169764	0.544458264	0.081392441	0.733014883	0.531368855	0
15	0.172529525	0.013556374	0.623446508	0.519135859	0.254456363	0.366947346	0.055695644	0.51850145	0.371256629	0.853783565	0.347039855	0.947346732	0.597281032	0.428806485	0.401369982	0.82044557	0.599584286	0.135680676	1
16	0.396513559	0.20124125	0.96810764	0.836439982	0.609976638	0.244681982	0.788270514	0.356016875	0.125054	0.510102589	0.403965968	0.865611911	0.499515991	0.589648486	0.014997916	0.893354828	0.562122469	0.028448737	1
17	0.051620363	0.310912551	0.909292558	0.719117468	0.433167082	0.997141749	0.231381709	0.771588936	0.593069372	0.109138528	0.21511746	0.051819288	0.29352604	0.751845499	0.922480759	0.606573034	0.318208833	0.718853514	1
18	0.814356436	0.696773866	0.702882553	0.975635677	0.983484815	0.031136722	0.938889143	0.744846045	0.752253549	0.41252405	0.586656405	0.941652686	0.247466384	0.550210882	0.229890761	0.691478239	0.999305937	0.647063187	1
19	0.595624121	0.791531254	0.794122938	0.430667284	0.916778135	0.897785418	0.729778768	0.560203906	0.658952968	0.835251912	0.644758866	0.444219065	0.820039098	0.87805546	0.238453253	0.41353844	0.378726412	0.973049473	1
20	0.453287468	0.227814194	0.868963932	0.83954677	0.133349425	0.951339573	0.179080954	0.02315243	0.144846838	0.674269176	0.083272596	0.437239218	0.231382255	0.424553435	0.578621063	0.140284237	0.538908979	0.620391974	1
21	0.358677493	0.539763975	0.408204473	0.391341766	0.660984427	0.730276729	0.415556778	0.677969692	0.092242834	0.4576951	0.971049521	0.116668505	0.893424314	0.895280358	0.41210954	0.067489641	0.254949351	0.146701198	1
22	0.795198342	0.129786545	0.881824612	0.531266301	0.546996702	0.62659594	0.689773244	0.941651133	0.466418688	0.098313262	0.435143216	0.730365887	0.868267937	0.852819668	0.525977488	0.785420605	0.630602682	0.740922908	1
23	0.500682859	0.360446702	0.939109798	0.149994564	0.579039918	0.297490247	0.905608324	0.765232184	0.026878729	0.608748029	0.316248889	0.842602638	0.384431766	0.600999813	0.94148821	0.981674888	0.225052715	0.69264385	1
24	0.029453696	0.675724624	0.024930062	0.1138323	0.833685553	0.268191042	0.936407155	0.577268761	0.728362992	0.846493622	0.41984847	0.286227754	0.782013775	0.455999155	0.450856709	0.909224239	0.827097995	0.125961426	1
25	0.294863722	0.817869572	0.548480255	0.983250152	0.47516934	0.672254588	0.198054688	0.270249808	0.453984699	0.725451126	0.653350694	0.099092477	0.369055732	0.252057032	0.426082946	0.381707776	0.775820591	0.56183777	1
26	0.483920717	0.272825273	0.288013515	0.424075319	0.825866787	0.686728398	0.317001138	0.107175529	0.761053776	0.720972963	0.984838874	0.345239468	0.145046735	0.961694304	0.021560996	0.127715148	0.978176445	0.877524397	1
27	0.773996627	0.029994884	0.500944728	0.181440099	0.501907079	0.017164768	0.683837319	0.239435751	0.694846243	0.762668235	0.118619659	0.694098152	0.489322637	0.35412897	0.527805905	0.77771341	0.309117119	0.262728187	1
28	0.420277396	0.371871862	0.74221177	0.063045151	0.189159879	0.8461391	0.874233445	0.143819031	0.156295523	0.35383536	0.393120692	0.633843456	0.788371192	0.856284486	0.602018601	0.168630821	0.53452174	0.987512719	1
29	0.531351045	0.148787195	0.345090002	0.085156677	0.737006982	0.345312469	0.172234035	0.690670496	0.097401026	0.183404353	0.371573714	0.480700616	0.506994244	0.823968274	0.146707842	0.595529737	0.371591388	0.199845544	1
30	0.305038056	0.240670239	0.830738987	0.021990788	0.998549574	0.294418373	0.716169725	0.415549576	0.921414983	0.139467971	0.069255819	0.981447689	0.20587566	0.194218463	0.50113342	0.181757552	0.691083869	0.615791178	1
31	0.250101395	0.734771845	0.937692761	0.892969553	0.235561915	0.982420485	0.532902829	0.398451868	0.195629774	0.898278459	0.73603117	0.889063171	0.978484093	0.828647165	0.442606477	0.452043046	0.769883358	0.43586573	1
32	0.384082258	0.883148776	0.029280328	0.359466858	0.357381684	0.52626826	0.036277993	0.590754629	0.137505628	0.048150918	0.662520833	0.338001542	0.30589365	0.02676161	0.030902354	0.339045275	0.576479067	0.759567056	1
33	0.715208537	0.329158344	0.576084515	0.022301076	0.384342267	0.822693803	0.716930135	0.325197325	0.854683674	0.39863879	0.835672806	0.467984427	0.004825306	0.682274386	0.11423621	0.566027905	0.559204189	0.802771117	1
34	0.678596222	0.115738127	0.529137065	0.188375472	0.41822212	0.333327638	0.138330019	0.120200379	0.532580008	0.967350228	0.158751384	0.474753502	0.142947952	0.16427107	0.809067914	0.768809702	0.93603262	0.402335295	1
35	0.742689086	0.420645437	0.431974905	0.6862906	0.043077611	0.041824483	0.986827916	0.868410077	0.414101277	0.530653655	0.422670447	0.182382541	0.897420062	0.72972669	0.158203094	0.875170029	0.618953154	0.218368287	1
36	0.01469819	0.944565988	0.245755275	0.680600516	0.967000433	0.449789209	0.593077848	0.272489439	0.676483967	0.756457615	0.61986926	0.909250391	0.702105575	0.763603493	0.376942453	0.477158463	0.352573291	0.328101977	1
37	0.198483945	0.436048952	0.668261273	0.995954859	0.109088587	0.034916367	0.264479133	0.992528838	0.700669385	0.410696096	0.000418997	0.309554721	0.731630352	0.699516576	0.143323291	0.249720026	0.129110679	0.823390264	1
38	0.492837052	0.236896454	0.636875173	0.541799712	0.06886407	0.860347024	0.457340372	0.059165141	0.382443994	0.349005878	0.333492747	0.2980105	0.763592802	0.064541143	0.00073196	0.745185426	0.671101148	0.0056988	1
39	0.942360705	0.758718331	0.475841498	0.833072592	0.09324978	0.16102077	0.828598425	0.215000754	0.392581685	0.650205947	0.818059952	0.016790827	0.393926545	0.622842759	0.068229331	0.592252771	0.555532291	0.102812312	1
40	0.403572912	0.624181003	0.352266838	0.69197124	0.214773626	0.814583784	0.437887973	0.220244534	0.57482734	0.256850098	0.545554604	0.3600059	0.052521026	0.920705524	0.676870104	0.828808791	0.299336209	0.121719626	1
41	0.12287734	0.604438373	0.491998566	0.117148366	0.403813848	0.5914582	0.657343118	0.18231043	0.654877857	0.558981592	0.600618184	0.034929981	0.745948438	0.055227859	0.951102504	0.742371089	0.386519077	0.784525826	1
42	0.809336176	0.252251145	0.55405121	0.056498849	0.56924897	0.053863752	0.476065605	0.044142905	0.154593587	0.218627511	0.356989707	0.382902889	0.215716961	0.717704731	0.32463964	0.713245168	0.461034506	0.522124229	1
43	0.931779857	0.534487388	0.072350192	0.508043944	0.797350636	0.854941721	0.534399284	0.499421222	0.865816304	0.514790592	0.500726072	0.41041037	0.549086272	0.11790858	0.901308792	0.274064094	0.644323262	0.038789302	1
44	0.319991405	0.500185766	0.542541513	0.57893229	0.939126686	0.187563712	0.643975153	0.101767815	0.610325831	0.380909514	0.878104988	0.681669557	0.929239327	0.988241664	0.136341162	0.128085921	0.648092821	0.384532118	1
45	0.67557668	0.904870744	0.965466725	0.763904438	0.633558896	0.759305095	0.216829513	0.255340614	0.665202852	0.368364859	0.74790174	0.934943769	0.956876101	0.420295416	0.986997475	0.955677423	0.485108416	0.206474049	0
46	0.132895597	0.96158949	0.514782509	0.819531342	0.288644342	0.528931266	0.796683771	0.843941962	0.037139483	0.464196776	0.776284523	0.77523183	0.124201822	0.74023159	0.883947314	0.203020178	0.032277739	0.68873416	1
47	0.366113848	0.162144608	0.559230181	0.888008265	0.33493644	0.089915759	0.466294991	0.652752669	0.541212114	0.475613799	0.373546016	0.709623018	0.613064649	0.884660433	0.875652156	0.549356306	0.166464893	0.537765079	1
48	0.115795248	0.761214168	0.344103065	0.777035822	0.1584583	0.664698656	0.576138704	0.044686546	0.774181312	0.497616293	0.394574679	0.067844535	0.057784017	0.153044141	0.613849775	0.271658446	0.059970371	0.595262407	1

newpop2

Figure 1: Processed data ready to be imported with pandas

```python
import pandas as pd
import sklearn as sk
import numpy as np
import math
import random
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
from sklearn.datasets import load_svmlight_file
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn import preprocessing, cross_decomposition, neighbors
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import ExtraTreesClassifier

#x_train, y_train = load_svmlight_file("/home/fubunutu/PycharmProjects/midterm/newpop.csv")

df = pd.read_csv("/home/fubunutu/PycharmProjects/midterm/newpop2.csv")

#create array, x values all columns except outcome. Y values = outcome column boolean
x = np.array(df.drop(['outcome'], 1))
y = np.array(df['outcome'])

x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.15)
clf = neighbors.KNeighborsClassifier(n_neighbors=3)
clf.fit(x_train, y_train)

# creating odd list of K for KNN
myList = list(range(1,50))

# subsetting just the odd ones
neighbors = list(filter(lambda x: x % 2 != 0, myList))

# empty list that will hold cv scores
cv_scores = []

#15-fold cross validation comparing accuracy of k values

for k in neighbors:
    clf = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(clf, x_train, y_train, cv=15, scoring='accuracy')
    cv_scores.append(scores.mean())

# misclassification error
MSE = [1 - x for x in cv_scores]

# determining best k
optimal_k = neighbors[MSE.index(min(MSE))]
print("The optimal number of neighbors is %d" % optimal_k)

# plot misclassification error vs k
plt.plot(neighbors, MSE)
plt.xlabel('Number of Neighbors K')
plt.ylabel('Misclassification Error')
plt.show()
acclist = []
```

Figure 2: First half of code: Imports data→ creates train/test sets → finds k with highest accuracy.

```python
loops = 10000
loopstr = str(loops)
# i = 1
for i in range(loops):
    x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.15)
    clf = KNeighborsClassifier(n_neighbors=optimal_k)
    clf.fit(x_train, y_train)
    accuracy = clf.score(x_test, y_test)
    accstr = str("%.3f" % accuracy)
    # np.append(acclist, accuracy)
    acclist.append(accuracy)
    opt = str(optimal_k)
    # i += 1

   # print('The accuracy of the model on the test dataset is ' + accstr + '% with k=' + opt)

y_pred = clf.predict(x_test)
cm = confusion_matrix(y_test, y_pred)
# Show confusion matrix in a separate window

plt.matshow(cm)
plt.title('Confusion matrix')
plt.colorbar()
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.show()

model = ExtraTreesClassifier(n_estimators=1000)
model.fit(x, y)
labels = df.head(0)
importance =np.array(model.feature_importances_)



print("The importance (0-1) of each feature with regard to the outcome ")
print(importance)
print("The average accuracy from " + loopstr + " simulations is:")
print(sum(acclist)/len(acclist))
```

Figure 3: 2nd half of code: Create array of accuracy → run 10000 knn sims with optimal k → average accuracy

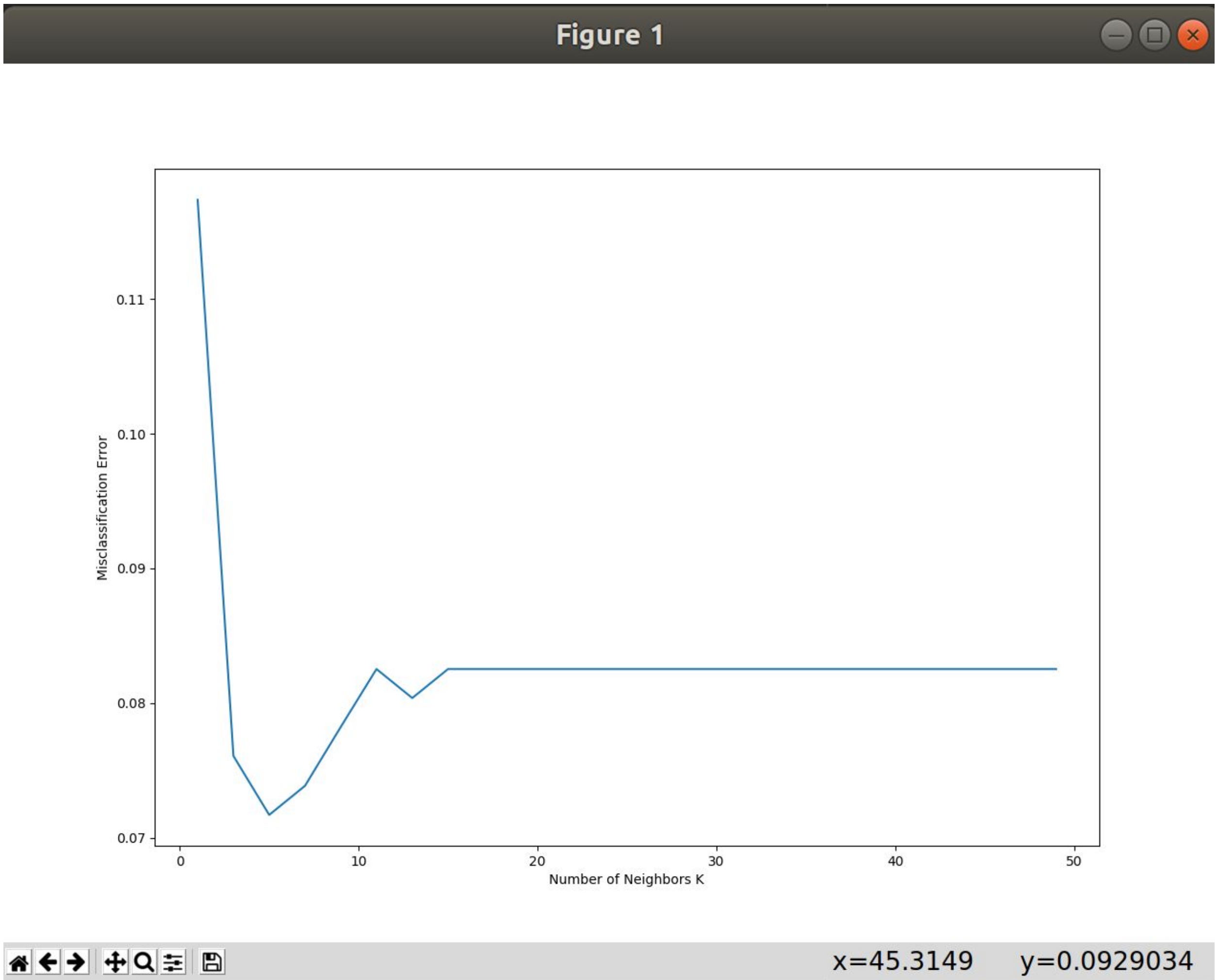

Figure 4: Plot of misclassification error for odd k values from 1-50

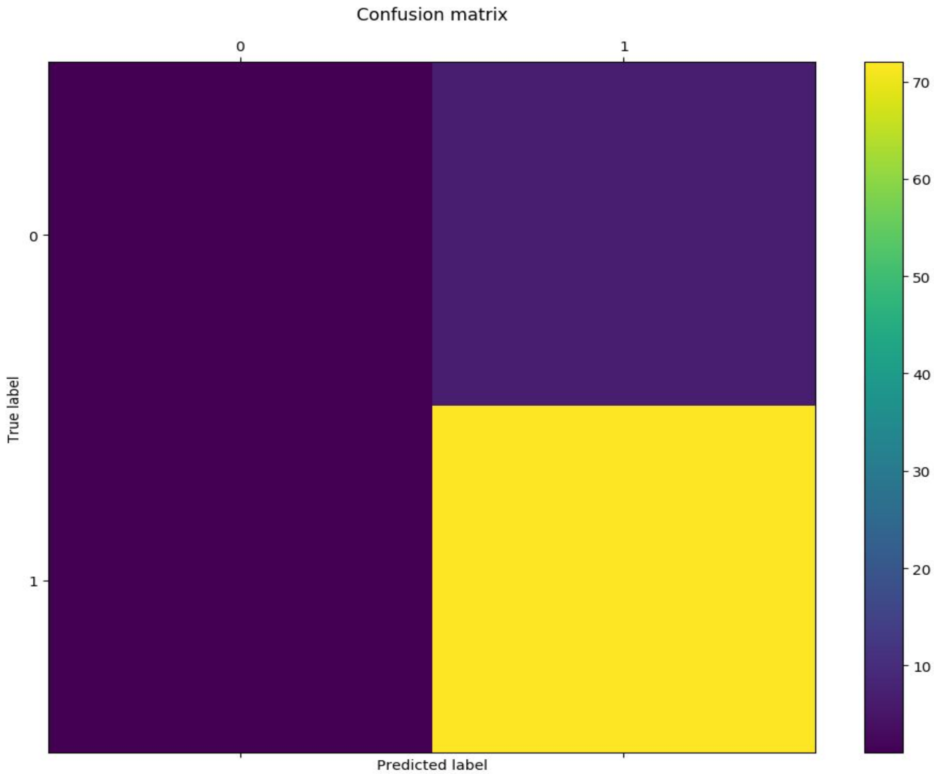

Figure 5: Confusion matrix showing the predicted vs true labels

```
/home/fubunutu/PycharmProjects/midterm/bin/python /home/fubunutu/Downloads/loopknn.py
The optimal number of neighbors is 5
The importance (0-1) of each feature with regard to the outcome
[0.1368148  0.13702292 0.03629747 0.04277402 0.04181272 0.03925524
 0.0363659  0.04143334 0.04415148 0.03823481 0.04223007 0.03887678
 0.08161605 0.07602488 0.04272585 0.04491572 0.03978081 0.03966712]
The average accuracy from 10000 simulations is:
0.9280506172839834

Process finished with exit code 0
```

Figure 6: Output: optimal_k, importance of each variable, and average accuracy