In [1]:
```
import pandas as pd
import numpy as np
import scipy.stats as scs
import statsmodels.api as sm
import matplotlib.pyplot as plt

%matplotlib inline
%config InlineBackend.figure_format='retina'
```

In [34]:
```
df = pd.read_csv('small_700_through_710_descr_clm_code.csv')
df.drop('Unnamed: 0',axis=1, inplace=True)
df = df[(df['code']==705)|(df['code']==706)|(df['code']==700)]
df['descr_clm'] = df.descr + df.clm
df.drop(['descr','clm'],axis=1, inplace=True)
df['code'] = df['code'].astype('category')
df.head()
```

Out[34]:

	code	descr_clm
0	700	This application claims priority under 35 U.S....
1	700	BACKGROUND \n 1. Field of Invention \n ...
2	700	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3	700	FIELD OF THE INVENTION \n The present inve...
4	700	RELATED APPLICATION \n This application cl...

In [35]:
```
df['code'].value_counts()
```

Out[35]:
```
706    1000
705    1000
700    1000
Name: code, dtype: int64
```

In [36]:
```
df['category_id'] = df['code'].factorize()[0]
```

In [37]:
```
df['category_id'].value_counts()
```

Out[37]:
```
1    1000
2    1000
0    1000
Name: category_id, dtype: int64
```

In [38]:
```
category_id_df = df[['code', 'category_id']].drop_duplicates().sort_valu
es('category_id')
category_to_id = dict(category_id_df.values)
id_to_category = dict(category_id_df[['category_id', 'code']].values)
```

In [39]:
```
id_to_category
```

Out[39]:
```
{0: 700, 1: 705, 2: 706}
```

Data exploration

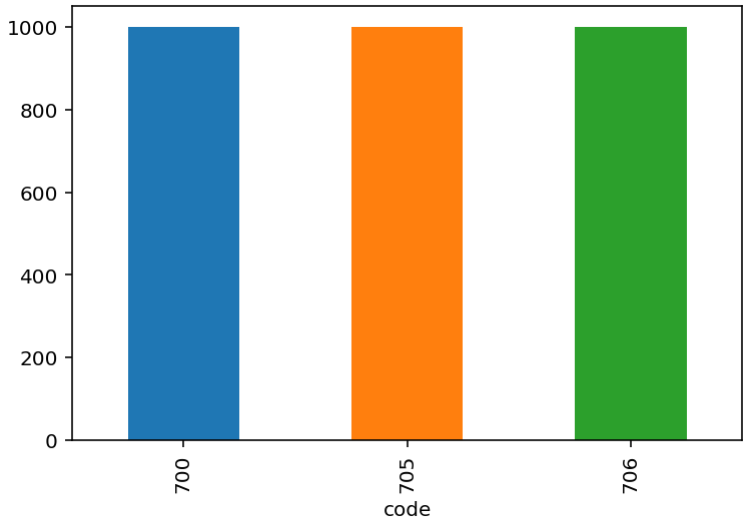
In [40]:
```
df.sample(5, random_state=0)
```

Out[40]:

	code	descr_clm	category_id
2077	700	CROSS-REFERENCE TO RELATED APPLICATION(S) \n ...	0
3401	706	BACKGROUND \n Many systems are instrumente...	2
4393	706	FIELD \n The subject matter disclosed here...	2
10648	705	This application is a continuation of U.S. pat...	1
10687	705	BACKGROUND \n 1. Technical Field \n Em...	1

In [41]:
```
df.groupby('code').descr_clm.count().plot.bar(ylim=0)
```

Out[41]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb38e580b00>
```



sklearn.feature_extraction.text.TfidfVectorizer will be used to calculate a tf-idf vector for each application.

In [42]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer(sublinear_tf=True, min_df=5, norm='l2', encoding
='latin-1', ngram_range=(1, 2), stop_words='english')

features = tfidf.fit_transform(df['descr_clm']).toarray()
labels = df['category_id']
features.shape
```

Out[42]:
```
(3000, 110325)
```

The 3000 applications are now each represented by 110325 features, representing the tf-idf score for different unigrams and bigrams.

In [43]:
```
from sklearn.feature_selection import chi2
```

In [46]:
```
N = 3
for code, category_id in sorted(category_to_id.items()):
  features_chi2 = chi2(features, labels == category_id)
  indices = np.argsort(features_chi2[0])
  feature_names = np.array(tfidf.get_feature_names())[indices]
  unigrams = [v for v in feature_names if len(v.split(' ')) == 1]
  bigrams = [v for v in feature_names if len(v.split(' ')) == 2]
  print("# '{}':".format(code))
  print("  . Most correlated unigrams:\n       . {}".format('\n       .
 '.join(unigrams[-N:])))
  print("  . Most correlated bigrams:\n       . {}".format('\n       . '
.join(bigrams[-N:])))
```

```
# '700':
  . Most correlated unigrams:
       . robot
       . control
       . controller
  . Most correlated bigrams:
       . control unit
       . perspective view
       . control device
# '705':
  . Most correlated unigrams:
       . merchant
       . transaction
       . payment
  . Most correlated bigrams:
       . service provider
       . credit card
       . point sale
# '706':
  . Most correlated unigrams:
       . neural
       . training
       . learning
  . Most correlated bigrams:
       . training data
       . machine learning
       . neural network
```

In [47]:
```
N = 10
for code, category_id in sorted(category_to_id.items()):
  features_chi2 = chi2(features, labels == category_id)
  indices = np.argsort(features_chi2[0])
  feature_names = np.array(tfidf.get_feature_names())[indices]
  unigrams = [v for v in feature_names if len(v.split(' ')) == 1]
  bigrams = [v for v in feature_names if len(v.split(' ')) == 2]
  print("# '{}':".format(code))
  print("  . Most correlated unigrams:\n       . {}".format('\n       .
 '.join(unigrams[-N:])))
  print("  . Most correlated bigrams:\n       . {}".format('\n       . '
.join(bigrams[-N:])))
```

```
# '700':
  . Most correlated unigrams:
       . temperature
       . direction
       . motor
       . position
       . sensor
       . controlling
       . power
       . robot
       . control
       . controller
  . Most correlated bigrams:
       . electric power
       . control signal
       . robot according
       . power supply
       . control method
       . method controlling
       . controller configured
       . control unit
       . perspective view
       . control device
# '705':
  . Most correlated unigrams:
       . card
       . price
       . sale
       . financial
       . transactions
       . credit
       . purchase
       . merchant
       . transaction
       . payment
  . Most correlated bigrams:
       . debit card
       . financial transaction
       . account number
       . payment transaction
       . mobile device
       . account associated
       . goods services
       . service provider
       . credit card
       . point sale
# '706':
  . Most correlated unigrams:
       . model
       . classifier
       . neurons
       . neuron
       . artificial
       . prediction
       . probability
       . neural
       . training
```

```
        . learning
    . Most correlated bigrams:
        . learning algorithm
        . model based
        . training set
        . artificial neural
        . knowledge base
        . artificial intelligence
        . neural networks
        . training data
        . machine learning
        . neural network
```

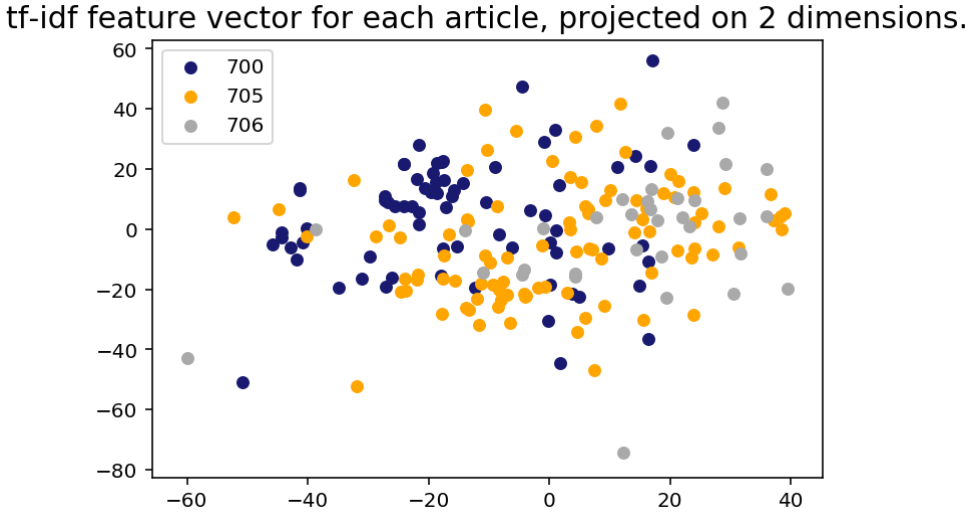
In [48]:
```
from sklearn.manifold import TSNE
```

In [50]:
```
# Sampling a subset of our dataset because t-SNE is computationally expe
nsive
SAMPLE_SIZE = int(len(features) * 0.3)
np.random.seed(0)
indices = np.random.choice(range(len(features)), size=SAMPLE_SIZE, repla
ce=False)
projected_features = TSNE(n_components=2, random_state=0).fit_transform(
features[indices])
colors = ['midnightblue', 'orange', 'darkgrey']
for category, category_id in sorted(category_to_id.items()):
    points = projected_features[(labels[indices] == category_id).values]
    plt.scatter(points[:, 0], points[:, 1], s=30, c=colors[category_id],
label=category)
plt.title("tf-idf feature vector for each article, projected on 2 dimens
ions.",
          fontdict=dict(fontsize=15))
plt.legend()
```

```
/home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages/panda
s/core/series.py:696: FutureWarning:
Passing list-likes to .loc or [] with any missing label will raise
KeyError in the future, you can use .reindex() as an alternative.

See the documentation here:
http://pandas.pydata.org/pandas-docs/stable/indexing.html#deprecate-loc
-reindex-listlike
  return self.loc[key]
```

Out[50]: <matplotlib.legend.Legend at 0x7fb375960d30>



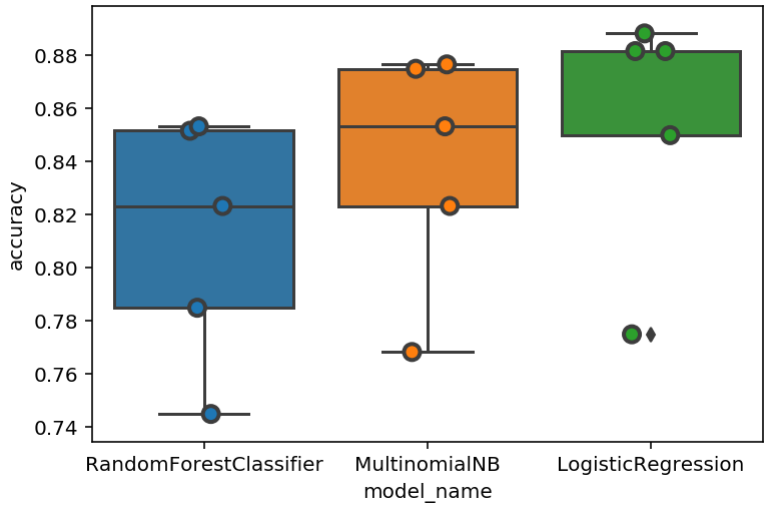
Model training and evaluation

In [51]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import cross_val_score
```

In [52]:
```
models = [
    RandomForestClassifier(n_estimators=200, max_depth=3, random_state=0
),
    MultinomialNB(),
    LogisticRegression(random_state=0),
]
CV = 5
cv_df = pd.DataFrame(index=range(CV * len(models)))
entries = []
for model in models:
  model_name = model.__class__.__name__
  accuracies = cross_val_score(model, features, labels, scoring='accurac
y', cv=CV)
  for fold_idx, accuracy in enumerate(accuracies):
    entries.append((model_name, fold_idx, accuracy))
cv_df = pd.DataFrame(entries, columns=['model_name', 'fold_idx', 'accura
cy'])
```

In [54]:
```
import seaborn as sns
```

In [55]:
```
sns.boxplot(x='model_name', y='accuracy', data=cv_df)
sns.stripplot(x='model_name', y='accuracy', data=cv_df,
              size=8, jitter=True, edgecolor="gray", linewidth=2);
```



In [56]:
```
cv_df.groupby('model_name').accuracy.mean()
```

Out[56]:
```
model_name
LogisticRegression        0.855333
MultinomialNB             0.839333
RandomForestClassifier    0.811667
Name: accuracy, dtype: float64
```

Model interpretation

In [57]:
```
from sklearn.model_selection import train_test_split
```

In [76]:
```
model = LogisticRegression(random_state=0)

X_train, X_test, y_train, y_test, indices_train, indices_test = train_te
st_split(features, labels, df.index, test_size=0.33, random_state=0, str
atify=df['code'])

model.fit(X_train, y_train)
y_pred_proba = model.predict_proba(X_test)
y_pred = model.predict(X_test)
```

In [77]:
```
y_train.value_counts()
```

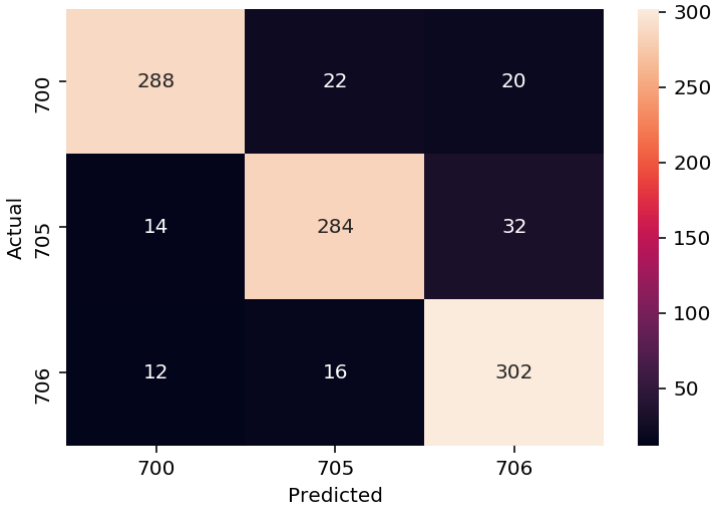
Out[77]:
```
2    670
1    670
0    670
Name: category_id, dtype: int64
```

In [79]:
```
y_test.value_counts()
```

Out[79]:
```
2    330
1    330
0    330
Name: category_id, dtype: int64
```

In [80]:
```
from sklearn.metrics import confusion_matrix
```

In [81]:
```
conf_mat = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_mat, annot=True, fmt='d',
            xticklabels=category_id_df.code.values, yticklabels=category
_id_df.code.values)
plt.ylabel('Actual')
plt.xlabel('Predicted');
```



In [82]:
```
from IPython.display import display

for predicted in category_id_df.category_id:
  for actual in category_id_df.category_id:
    if predicted != actual and conf_mat[actual, predicted] >= 2:
      print("'{}' predicted as '{}' : {} examples.".format(id_to_categor
y[actual], id_to_category[predicted], conf_mat[actual, predicted]))
      display(df.loc[indices_test[(y_test == actual) & (y_pred == predic
ted)]][['descr_clm']])
      print('')
```

'705' predicted as '700' : 14 examples.

	descr_clm
1902	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
10656	TECHNICAL FIELD \n The present invention r...
558	CROSS-REFERENCES TO RELATED APPLICATIONS \n ...
532	CROSS-REFERENCE TO RELATED PATENT APPLICATION ...
1896	CROSS REFERENCE TO RELATED APPLICATIONS \n ...
2345	FIELD OF THE INVENTION \n Embodiments gene...
2378	BACKGROUND OF THE INVENTION \n 1. Field of...
7992	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
4298	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
1872	This application claims priority to U.S. Provi...
8021	RELATED PATENT APPLICATIONS \n This applic...
4229	RELATED APPLICATIONS \n The present invent...
8083	BACKGROUND \n Entities such as data center...
8106	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...

'706' predicted as '700' : 12 examples.

	descr_clm
4704	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
2572	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3105	BACKGROUND \n Earth formations may be used...
4630	CROSS-REFERENCES TO RELATED APPLICATIONS \n ...
3356	RELATED APPLICATION INFORMATION \n This ap...
3360	BACKGROUND OF INVENTION \n 1. Field of the...
4373	RELATED APPLICATION INFORMATION \n The pre...
3345	CLAIM FOR PRIORITY \n This application cla...
4481	RELATED APPLICATIONS \n This application c...
4607	CROSS REFERENCE TO RELATED APPLICATIONS \n ...
3303	FIELD AND BACKGROUND OF THE INVENTION \n T...
3351	TECHNICAL FIELD \n The present invention r...

'700' predicted as '705' : 22 examples.

	descr_clm
3972	This application claims the benefit of U.S. pr...
743	CROSS REFERENCE TO RELATED APPLICATIONS \n ...
3707	CROSS REFERENCE TO RELATED APPLICATIONS \n ...
3850	This application claims the benefit of U.S. Pr...
757	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3706	BACKGROUND \n Shipments of items are often...
3694	CROSS REFERENCE TO RELATED APPLICATIONS \n ...
6112	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3748	CLAIM OF BENEFIT TO PRIOR APPLICATION \n T...
3761	CROSS REFERENCE TO RELATED APPLICATIONS \n ...
3700	BACKGROUND \n Retailers, wholesalers, and ...
3971	TECHNICAL FIELD \n This disclosure relates...
760	CROSS-REFERENCE TO RELATED APPLICATION \n ...
5149	PRIORITY CLAIM \n This application is a co...
788	BACKGROUND \n Three-dimensional (3D) print...
5186	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
5129	BACKGROUND \n Shipments of items are often...
2074	This utility patent application is a continuat...
2084	PRIORITY APPLICATION \n This application i...
5191	The current application claims a priority to t...
3746	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3710	CROSS REFERENCE TO RELATED APPLICATION \n ...

```
'706' predicted as '705' : 16 examples.
```

	descr_clm
3181	RELATED APPLICATIONS \n This application i...
3404	CLAIM OF PRIORITY \n This application clai...
3354	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3176	BACKGROUND \n 1. Field \n The system a...
4693	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3099	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
4430	FIELD OF THE DISCLOSURE \n The present dis...
3298	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3304	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
3287	CONTINUATION-IN-PART \n This application i...
3174	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
4656	TECHNICAL FIELD \n The present disclosure ...
2554	RELATED APPLICATIONS \n This application i...
594	CROSS REFERENCE TO OTHER APPLICATIONS \n T...
4396	CROSS REFERENCE TO RELATED APPLICATION \n ...
4491	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...

```
'700' predicted as '706' : 20 examples.
```

	descr_clm
5227	RELATED APPLICATIONS \n This application c...
784	TECHNICAL FIELD \n The present disclosure ...
2076	CLAIM OF PRIORITY \n This application is a...
2092	CROSS REFERENCE TO RELATED APPLICATIONS \n ...
2117	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
2090	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
2107	CROSS REFERENCE TO RELATED APPLICATION \n ...
5312	FIELD OF THE INVENTION \n This description...
5318	RELATED APPLICATIONS \n This application c...
2036	FIELD \n Embodiments taught herein relate ...
3641	FIELD OF THE INVENTION \n The present inve...
5061	FIELD OF THE INVENTION \n The present inve...
5082	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
5072	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
2085	TECHNICAL FIELD \n This disclosure relates...
5300	REFERENCE TO RELATED APPLICATION \n This a...
3961	TECHNICAL FIELD \n The present disclosure ...
774	RELATED APPLICATIONS \n This application c...
5341	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
5083	TECHNICAL FIELD \n Embodiments of the subj...

```
'705' predicted as '706' : 32 examples.
```

	descr_clm
7101	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
10682	BACKGROUND \n Advertisers increasingly see...
7030	This application is a continuation of U.S. Ser...
7113	BACKGROUND \n Product lifecycle management...
7095	FIELD OF THE INVENTION \n The present inve...
8074	RELATED APPLICATIONS \n This application i...
8069	TECHNICAL FIELD \n The present disclosure ...
7029	TECHNICAL FIELD \n Embodiments of the pres...
7039	CLAIM OF PRIORITY \n This application is a...
530	This application claims the benefit of U.S. Pr...
8042	REFERENCE TO RELATED APPLICATIONS \n This ...
9311	RELATED APPLICATIONS \n This patent applic...
542	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
1903	TECHNICAL FIELD OF THE INVENTION \n The in...
572	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
7034	RELATED APPLICATIONS \n This application c...
10632	TECHNICAL FIELD \n The present technology ...
10643	PRIORITY \n This application claims priori...
9356	TECHNICAL FIELD \n The present disclosure ...
9327	CROSS REFERENCE TO OTHER APPLICATIONS \n T...
8046	PRIORITY DATA \n This application claims p...
9344	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...
7105	BACKGROUND \n This disclosure relates gene...
7112	BACKGROUND \n Online social networks allow...
562	BACKGROUND \n Unless otherwise indicated h...
7119	CROSS REFERENCE TO RELATED APPLICATIONS \n ...
4255	This application is a Continuation of U.S. app...
7972	FIELD OF THE INVENTION \n The present inve...
9335	This application relates to U.S. provisional p...
566	TECHNICAL FIELD \n The present invention r...
9349	CROSS REFERENCE TO RELATED APPLICATION \n ...
8044	CROSS-REFERENCE TO RELATED APPLICATIONS \n ...

In [83]:
```
model.fit(features, labels)
```

Out[83]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=
True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=
1,
          penalty='l2', random_state=0, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

In [84]:
```
from sklearn.feature_selection import chi2

N = 10
for category, category_id in sorted(category_to_id.items()):
  indices = np.argsort(model.coef_[category_id])
  feature_names = np.array(tfidf.get_feature_names())[indices]
  unigrams = [v for v in reversed(feature_names) if len(v.split(' ')) ==
1][:N]
  bigrams = [v for v in reversed(feature_names) if len(v.split(' ')) ==
2][:N]
  print("# '{}':".format(category))
  print("  . Top unigrams:\n       . {}".format('\n       . '.join(unigr
ams)))
  print("  . Top bigrams:\n       . {}".format('\n       . '.join(bigram
s)))
```

```
# '700':
  . Top unigrams:
       . control
       . controller
       . power
       . controlling
       . robot
       . sensor
       . position
       . energy
       . temperature
       . operation
  . Top bigrams:
       . control device
       . controller configured
       . perspective view
       . method controlling
       . control unit
       . control method
       . electric power
       . power supply
       . control signal
       . control systems
# '705':
  . Top unigrams:
       . payment
       . transaction
       . price
       . customer
       . financial
       . purchase
       . transactions
       . provider
       . service
       . account
  . Top bigrams:
       . service provider
       . mobile device
       . point sale
       . method providing
       . readable medium
       . implemented method
       . method claim
       . credit card
       . goods services
       . information associated
# '706':
  . Top unigrams:
       . learning
       . probability
       . training
       . rule
       . model
       . prediction
       . neural
       . predicting
       . knowledge
```

```
        . artificial
    . Top bigrams:
        . neural network
        . machine learning
        . training data
        . knowledge base
        . artificial intelligence
        . input data
        . readable storage
        . neural networks
        . artificial neural
        . data set
```

In [85]:
```
df[df.descr_clm.str.lower().str.contains('machine learning')].code.value_counts()
```

Out[85]:
```
706    178
705     17
700      8
Name: code, dtype: int64
```

In [86]:
```
texts = ["1. A system comprising:a memory that stores computer executabl
e components a processor that executes computer executable components st
ored in the memory wherein the computer executable components comprise:a
snapshot component that generates a first sequence of multi-dimensional
 time series data and a second sequence of multi-dimensional time series
data from multi-dimensional time series data associated with at least tw
o different data types generated by a data system over a consecutive per
iod of time and a machine learning component that analyzes the first seq
uence of multi-dimensional time series data and the second sequence of m
ulti-dimensional time series data using a convolutional neuralnetwork sy
stem to predict an event associated with the multi-dimensional time seri
es data. 2. The system of claim 1 wherein the snapshot component generat
es a data matrix associated with the first sequence of multi-dimensional
time series data and the second sequence of multi-dimensional time serie
s data and wherein the machine learning component analyzes the data matr
ix using the convolutional neural network system. 3. The system of claim
1 wherein the machine learning component analyzes the first sequence of
 multi-dimensional time series data and the second sequence of multi-dim
ensional time series data using a parallel network of processing units a
ssociated with the convolutional neural network system and wherein perfo
rmance of the processor to predict the event associated with the multi-d
imensional time series data is improved by employing the convolutional n
eural network system. 4. The system of claim 1 wherein the snapshot comp
onent determines a size of the first sequence of multi-dimensional time
 series data and the second sequence of multi-dimensional time series da
ta based on data associated with the convolutional neural network syste
m. 5. The system of claim 1 wherein the snapshot component determines a
 set of parameters for the convolutional neuralnetwork system based on a
classification of data associated with the convolutional neural networ
k system. 6. The system of claim 1 wherein a portion of data from the fi
rst sequence of multi-dimensional time series data corresponds to the se
cond sequence of multi-dimensional time series data and wherein the mach
ine learning component analyzes the portion of the data. 7. The system o
f claim 1 wherein at least the first sequence of multi-dimensional time
 series data comprises dynamic data that is related to other data includ
ed in the first sequence of multi-dimensional time series data or the se
cond sequence of multi-dimensional time series data and wherein the mach
ine learning component analyzes the dynamic data. 8. The system of claim
1 wherein the snapshot component generates the first sequence of multi-d
imensional time series data and the second sequence of multi-dimensional
time series data based on feedback data indicative of information for tu
ning the first sequence of multi-dimensional time series data and the se
cond sequence of multi-dimensional time series data. 9. The system of cl
aim 1 wherein the machine learning component adjusts the convolutional n
eural network system based on feedback data indicative of information fo
r tuning the convolutional neural network system. 10. The system of clai
m 1 wherein the convolutional neuralnetwork system is associated with no
nlinear processing of features associated with the first sequence of mul
ti-dimensional time series data and the second sequence of multi-dimensi
onal time series data. 11. The system of claim 1 further comprising:a di
splay component that generates a user interface to display output data a
ssociated with the event in a human interpretable format. 12-19. (cancel
ed) 20. A computer program product for machine learning the computer pro
gram product comprising a computer readable storage medium having progra
m instructions embodied therewith the program instructions executable by
processor to cause the processor to:generate by the processor a data mat
```

```
rix based on first time series data associated with a first data source
 and second time series data associated with a second data source analyz
e by the processor the data matrix associated with the first time series
data and the second time series data using a convolutional neural networ
k system and generate by the processor prediction data that comprises a
 predicted event associated with the first time series data and the seco
nd time series data based on data generated by the convolutional neura
l network system. 21. The computer program product of claim 20 wherein t
he program instructions are further executable by the processor to cause
the processor to:modify by the processor the data matrix based on the da
ta generated by the convolutional neural network system. 22. The compute
r program product of claim 20 wherein the program instructions are furth
er executable by the processor to cause the processor to:modify by the p
rocessor a convolutional neural networkassociated with the convolutiona
l neural network system based on the data generated by the convolutiona
l neuralnetwork system. 23. A computer program product for machine learn
ing the computer program product comprising a computer readable storage
 medium having program instructions embodied therewith the program instr
uctions executable by processor to cause the processor to:generate by th
e processor a data matrix based on multi-dimensional time series data as
sociated with at least two different data types perform by the processor
a convolutional neural network process based on the data matrix associat
ed with the multi-dimensional time series data and generate by the proce
ssor prediction data that comprises a predicted event associated with th
e multi-dimensional time series data based on the convolutional neural n
etwork process. 24. The computer program product of claim 23 wherein the
program instructions are further executable by the processor to cause th
e processor to:modify by the processor the data matrix based on the pred
iction data. 25. The computer program product of claim 23 wherein the pr
ogram instructions are further executable by the processor to cause the
 processor to:modify by the processor a convolutional neural networkasso
ciated with the convolutional neural network process based on the predic
tion data."]
text_features = tfidf.transform(texts)
predictions = model.predict(text_features)
for text, predicted in zip(texts, predictions):
  print('"{}"'.format(text))
  print("  - Predicted as: '{}'".format(id_to_category[predicted]))
  print("")
```

"1. A system comprising:a memory that stores computer executable compon ents a processor that executes computer executable components stored in the memory wherein the computer executable components comprise:a snapsh ot component that generates a first sequence of multi-dimensional time series data and a second sequence of multi-dimensional time series data from multi-dimensional time series data associated with at least two di fferent data types generated by a data system over a consecutive period of time and a machine learning component that analyzes the first sequen ce of multi-dimensional time series data and the second sequence of mul ti-dimensional time series data using a convolutional neuralnetwork sys tem to predict an event associated with the multi-dimensional time seri es data. 2. The system of claim 1 wherein the snapshot component genera tes a data matrix associated with the first sequence of multi-dimension al time series data and the second sequence of multi-dimensional time s eries data and wherein the machine learning component analyzes the data matrix using the convolutional neural network system. 3. The system o f claim 1 wherein the machine learning component analyzes the first seq uence of multi-dimensional time series data and the second sequence of multi-dimensional time series data using a parallel network of processi ng units associated with the convolutional neural network system and wh erein performance of the processor to predict the event associated with the multi-dimensional time series data is improved by employing the con volutional neural network system. 4. The system of claim 1 wherein the snapshot component determines a size of the first sequence of multi-dim ensional time series data and the second sequence of multi-dimensional time series data based on data associated with the convolutional neura l network system. 5. The system of claim 1 wherein the snapshot compone nt determines a set of parameters for the convolutional neuralnetwork s ystem based on a classification of data associated with the convolution al neural network system. 6. The system of claim 1 wherein a portion of data from the first sequence of multi-dimensional time series data corr esponds to the second sequence of multi-dimensional time series data an d wherein the machine learning component analyzes the portion of the da ta. 7. The system of claim 1 wherein at least the first sequence of mul ti-dimensional time series data comprises dynamic data that is related to other data included in the first sequence of multi-dimensional time series data or the second sequence of multi-dimensional time series dat a and wherein the machine learning component analyzes the dynamic data. 8. The system of claim 1 wherein the snapshot component generates the f irst sequence of multi-dimensional time series data and the second sequ ence of multi-dimensional time series data based on feedback data indic ative of information for tuning the first sequence of multi-dimensional time series data and the second sequence of multi-dimensional time seri es data. 9. The system of claim 1 wherein the machine learning componen t adjusts the convolutional neural network system based on feedback dat a indicative of information for tuning the convolutional neural networ k system. 10. The system of claim 1 wherein the convolutional neuralnet work system is associated with nonlinear processing of features associa ted with the first sequence of multi-dimensional time series data and t he second sequence of multi-dimensional time series data. 11. The syste m of claim 1 further comprising:a display component that generates a us er interface to display output data associated with the event in a huma n interpretable format. 12-19. (canceled) 20. A computer program produc t for machine learning the computer program product comprising a comput er readable storage medium having program instructions embodied therewi th the program instructions executable by processor to cause the proces sor to:generate by the processor a data matrix based on first time seri

```
es data associated with a first data source and second time series data
associated with a second data source analyze by the processor the data
matrix associated with the first time series data and the second time s
eries data using a convolutional neural network system and generate by
the processor prediction data that comprises a predicted event associat
ed with the first time series data and the second time series data base
d on data generated by the convolutional neural network system. 21. The
computer program product of claim 20 wherein the program instructions a
re further executable by the processor to cause the processor to:modify
by the processor the data matrix based on the data generated by the con
volutional neural network system. 22. The computer program product of c
laim 20 wherein the program instructions are further executable by the
processor to cause the processor to:modify by the processor a convoluti
onal neural networkassociated with the convolutional neural network sys
tem based on the data generated by the convolutional neuralnetwork syst
em. 23. A computer program product for machine learning the computer pr
ogram product comprising a computer readable storage medium having prog
ram instructions embodied therewith the program instructions executable
by processor to cause the processor to:generate by the processor a data
matrix based on multi-dimensional time series data associated with at l
east two different data types perform by the processor a convolutiona
l neural network process based on the data matrix associated with the m
ulti-dimensional time series data and generate by the processor predict
ion data that comprises a predicted event associated with the multi-dim
ensional time series data based on the convolutional neural network pro
cess. 24. The computer program product of claim 23 wherein the program
instructions are further executable by the processor to cause the proce
ssor to:modify by the processor the data matrix based on the prediction
data. 25. The computer program product of claim 23 wherein the program
instructions are further executable by the processor to cause the proce
ssor to:modify by the processor a convolutional neural networkassociate
d with the convolutional neural network process based on the prediction
data."
  - Predicted as: '706'
```

https://patentimages.storage.googleapis.com/e2/3f/75/2f32412145e159/US20180260697A1.pdf
(https://patentimages.storage.googleapis.com/e2/3f/75/2f32412145e159/US20180260697A1.pdf)